NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

**A REACTIVE TARGET
ACTIVE ASW SONAR SEARCH
TACTICAL DECISION AID**

by

Cesar J. Recalde

September, 1996

Thesis Advisor: Alan R. Washburn

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE September 1996.	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE A REACTIVE TARGET ACTIVE ASW SONAR SEARCH TACTICAL DECISION AID.	5. FUNDING NUMBERS
6. AUTHOR(S) Cesar Julio Recalde	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.	12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

This thesis develops, implements and tests a Tactical Decision Aid for a Reactive Target ASW Active Search. The model uses a Bayesian Filtering Process to fuse information from a real world search conducted by several assets with information from a Monte Carlo Simulation that encompasses five hundred equally likely different possible initial positions and behaviors of the real target. A Reactive Target Model resembles the behavior of a target that is always aware and reacts because of the presence and activity of the searchers. An initial "prior", or best estimate of the location of the target is updated using the movement of the simulated targets, the negative information conveyed in an unsuccesful search over a period of time and the positive information implied in a contact report. The search effort is measured using a Fixed Scan Stochastic Model that solves the Sonar Equation limited by noise and reverberation. As a result of updating the prior, a "posterior" distribution is obtained. The Law of Total Probabilities is used to render a probability map of the location of the Target by mapping color intensities to probabilities. A recursive expression for evaluating a contact report is also developed.

14. SUBJECT TERMS Search, Detection, Fusion, Sonar, Tactics, Antisubmarine.			15. NUMBER OF PAGES 165
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)

Prescribed by ANSI Std. 239-18 298-102

Approved for public release; distribution is unlimited.

A REACTIVE TARGET ACTIVE ASW SONAR SEARCH TACTICAL DECISION AID

Cesar J. Recalde
Lieutenant, Argentine Navy
Argentine Naval Academy, 1984.

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 1996

ABSTRACT

This thesis develops, implements and tests a Tactical Decision Aid for a Reactive Target ASW Active Search. The model uses a Bayesian Filtering Process to fuse information from a real world search conducted by several assets with information from a Monte Carlo Simulation that encompasses five hundred equally-likely different possible initial positions and behaviors of the real target. A Reactive Target Model resembles the behavior of a target that is always aware and reacts because of the presence and activity of the searchers. An initial "prior", or best estimate of the location of the target, is updated using the movement of the simulated targets, the negative information conveyed in an unsuccesful search over a period of time and the positive information implied in a contact report. The search effort is measured using a Fixed Scan Stochastic Model that solves the Sonar Equation limited by noise and. As a result of updating the prior, a "posterior" distribution is obtained. The Law of Total Probabilities is used to render a probability map of the location of the Target by mapping color intensities to probabilities. A recursive expression for evaluating a contact report is also developed.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

I am deeply grateful to the Argentine Navy and the United States Navy for giving me the opportunity to live the rewarding experience of the Naval Postgraduate School and to the following individuals that made this thesis possible.

Professor James Eagle encouraged me to pursue an Operations Research degree from the Undersea Warfare Curriculum. Without his support I would have never endeavored such a challenge.

Professor Washburn was always a sturdy helmsman in the turbulent waters of probabilities. His rigor in pursuing the truth, albeit challenging and demanding, gave solid ground to my research and took this thesis safely to a secure port. I am greatly thankful for his forbearance.

Professor Sanders built a bridge between the Physics and the Operations Research Departments by helping me construct a simple yet veritable sound propagation model. I sincerely appreciate his help and support.

I am honored to acknowledge Professor Wayne P. Hughes for his comments on Chapter I. He provided me with the naval scope that gave sense to the idea.

I am profoundly thankful to Andrea, my wife, soul mate and best friend, who not only supported me during this two years but also motivated me towards what is right and better, even when it was harder for both of us. Her loving encouragement and understanding made everything possible, worthwhile, and unforgettable. Finally to Catalina, my daughter to-be, who knowingly must have delayed her arrival into this world as an expression of support for the signature of this thesis.

I. INTRODUCTION

A. BACKGROUND

The end of the cold war started a transition from the Major Regional Conflicts to the Lesser Intensity Conflicts (Hughes, 1993,pp.1) as the more likely scenario to be developed in the future for US Forces. Such a transition renewed the interest and efforts to study what is called "Littoral Warfare". Other allied countries have had the Littoral Warfare as their natural scope and permanent frame for the design and development of the capabilities of their naval forces.

Above all, Littoral Warfighting implies sensible changes in the physical nature of the properties of the threat and the environment which together with the mission and the current capabilities determines the imbalance that drives research, development and acquisition.

1. The environment in Littoral Warfare

The changes in the physical properties of the environment due to the transition from the Open Sea to Littoral or Coastal waters have already been thoroughly studied and are characterized by the distinction between deep and shallow waters, often termed blue and green waters respectively. A particular maritime area is considered a shallow water environment in terms of the influence its boundaries exert in its physical properties rather than in terms of a particular predetermined depth.

From the standpoint of the Search and Detection problem, which remains crucial to Littoral Warfighting (Hughes,1986,pp-126), the most remarkable differences between shallow or deep water environments are the intensity of noise and clutter and the spatial and temporal variability in the properties of the ocean which become distinctive and determinant factors affecting the performance of the sensors and the decision aids associated with them. Several factors have a direct impact in increasing the intensities of clutter and noise. The most important factors are the wind, the biological content of the water, the proximity of the bottom, and human activity.

The wind increases both ambient noise and clutter. It has been observed that for the typical frequencies used in passive sonar, the ambient noise in coastal locations is in general more intense in shallow waters. The simplest reason is the larger mass of absorption in the deep ocean. In shallow waters the noise produced by the wind (white caps, bubbles, waves, etc.) is partially reflected by the bottom and the surface, hence the dissipation process is slower. The increase in clutter is due to the roughness of the surface which in turn increases as the wave length decreases due to the reduction of depth (Urick,1975, pp. 236). Additionally, particular conditions of short waves typical of shallow waters like tide rips, choppy seas, etc., make detection harder due to the movement transmitted to the platforms where the sensors are carried.

Biological activity increases as depth decreases thus

leading to an increase in both noise (crabs, shrimps,etc.) and clutter (whales, schools of fish, zooplankton, etc.).(Urick, 1975,pp.193)

Human activity produces both noise and clutter. Shipping activity increases close to shore. A fishing vessel, for instance, can be considered a source of both noise and clutter.

Finally, the closer bottom of the ocean not only increases the backscattering due to reflection in the case of an active sensor, but also prevents the formation of caustics or convergence zones typical of deep waters.

Meteorological and Oceanographical changes also intensify when transitioning from the macroscale of the open ocean to the mesoscale of coastal regions. For instance, daily variations in the temperature profile in the water and in the air lead to drastic changes in the propagation of electromagnetic and acoustic energy.

The prediction of the performance of a sensor for a particular target and environment is of the essence in the searching problem. The more intense the variability in the conditions found in shallow or coastal waters the harder the prediction. Consequently, the uncertainty in the evaluation of the effectiveness of a given detection effort increases.

2. Changes in the threat

With the end of the cold war, the most likely threat

became smaller but not weaker, at least from a conventional warfare point of view. Small fast patrol boats can strike fast with remarkable lethality and are not easy to detect in choppy seas. Modern conventional submarines have the same or even better information systems than older nuclear submarines, taking advantage of multiple sensors and massive signal processing based on inexpensive and commercially available technology. Except for the snorkeling cycle where they become noisy and visible, conventional submarines on batteries are almost undetectable by current passive sensors. Their smaller size makes it more difficult to detect them with active sensors due to a reduction in target strength.

3. Sensors and Tactical Decision Aids(TDA)

The threat and the environment are determinant factors in the decision to use a particular type of sensor and whether it will be used in a passive or active fashion. Regardless of its type or mode, a sensor normally requires a Tactical Decision Aid for a more intelligent and efficient use.

In the realm of the ASW Search Problem, the fact that a diesel submarine on batteries is almost undetectable with passive acoustic means, especially in the noisy shallow water environment, drives the need to use active sensors. This conclusion can be extended to other threat-environment combinations in shallow waters, for instance small fast surface combatants because they normally operate in silence

unless they are about to strike.

A sensor typically feeds a Tactical Data System where software tools (TDA) help decide the best course of action given a particular profile of mission, environment and threat. The essence of a Tactical Decision Aid is providing qualitative and quantitative processing of the data, so it becomes information. The factors of the situation should be ranked, simplified and presented in a way such that the determinant ones are emphasized in terms of the mission.

The smaller signal to noise and target to clutter ratios due to the more noisy and cluttered environment will not only make detection much more difficult but also, together with strong environmental variability, will generate more uncertainty about the effectiveness of the search. As a consequence, the need for a stochastic TDA to keep adequate track of the uncertainties is evident. Figure 1.1 summarizes the problem structure leading to the development of a TDA. The chains of causes and effects are displayed as a flow pattern.

B. THE PROBLEM

For any Navy that does not have a TDA capable of handling the uncertainty, the problem is apparent and the need to develop one is clear.

A large array of TDA's in the U.S. NAVY like VPCAS, ASWTDA and PACSEARCH (Wagner, 1989,pp.II.2) responded to the needs of the cold war where the most important means of

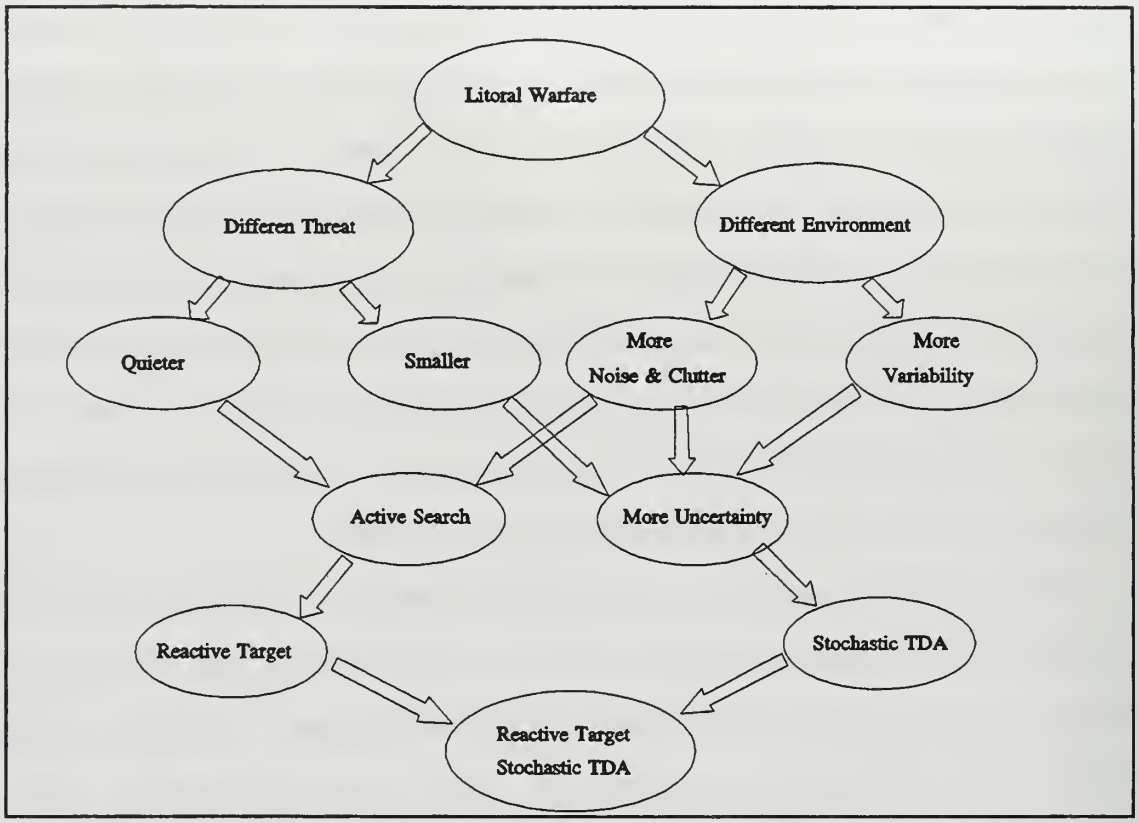

Figure 1.1. The structure of the problem of Active Search in Shallow water is remarkable different than Passive Search in Deep Water.

detection was the passive sonar. In a passive sonar search, specially from airborne platforms, the target will usually not be aware of the presence of the searchers. As a consequence those TDA's do not consider any reaction of the target caused by the presence or activities of the searcher.

In an active search, the target will almost invariably know of the presence, location and activities of the searcher before detection. A simple signal intensity budget shows that there exists a distance where the energy suffices to produce a detection in the passive receptor of the target, but it is not enough to produce the same effect at the receptor of the

active device in the searcher. Geometrical spreading and absorption in the medium account for the difference. The active searcher is thus a beacon with the choice of avoidance or approach up to the target. Bearings from the target to the searchers change noticeably due to the shorter distances and higher speed of the actions, thus enabling a relatively accurate Motion Analysis (Wagner, 1989,pp.III-2) by the target.

As a consequence, the target is not only aware of the presence of the searcher but also can have a very clear picture of the situation. It will invariably have a mission driven reaction, either to attack or evade. Information becomes crucial and the target has the advantage. This feature of the target is the essence of the problem, hence the name RTCAS, Reactive Target Computer Assisted Search.

Time and distance scales change as well when transitioning from a passive to an active search in the case of sonar. A passive search is a resource employed in a larger time frame and over a more extended area than the active search. Hence the name Area ASW for the actions typically carried out in the open ocean either to "sanitize" a corridor or to survey a given wide area for strategic purposes. Conversely, active search in ASW is a focal, more concentrated effort. The detection ranges are shorter hence the space scale of the search is smaller. The cost in terms of allocation of resources is higher in an active search, hence its sustainabi-

lity decreases and the time scales shrink. Minutes count.

C. APPROACHING A SOLUTION

The objective of this thesis is to develop a Reactive Target Tactical Decision Aid for Computer Assisted Search.

TDA's are aimed to assimilate and conveniently present the data (target, own assets and environment) to the decision maker, i.e., transform the data into information and "...analyze the tactical problem beyond what is possible by humans in a timely fashion..." (Wagner, 1989, pp.I.1). That analysis may either yield an evaluative picture from where the decision maker will construct a course of action more easily or produce a recommended course of action. The first step is always needed and crucial to the second, so this thesis is constrained to an evaluative TDA leaving the assessment capabilities as a future enhancement.

Given the uncertainties involved in the problem and following approximately the same approach as previous TDA's did, RTCAS uses a Monte Carlo Simulation to represent a Reactive Target, a Fixed Scan Model to measure the detection effort and a Bayesian Filtering Process to fuse the simulation with the real world search activity. This section describes those models.

1. The Information Fusion Paradigm

Figure 1.2 is a pictorial description of the information fusion paradigm applied to a search TDA. Two main sources of

information can be distinguished, reality and a Monte Carlo simulation. Reality provides the information from the tactical situation, the datum, the time, and searcher activity. The Monte Carlo Simulation provides a large number of possible locations and movement of the target based on the datum, each one equally likely, at least at the beginning, to be the real world target: the "prior."

Fusing the information consists of updating the "prior" or probability map of the location of the target using all available information. There are three main causes of updating, namely the passage of time, negative information and positive information. In the first case the prior will move with the variety of courses and speeds provided by the Monte Carlo Simulated Targets. A large number of simulated tracks (500 in RTCAS) resemble many of the possible positions and movements of the real target. If no searching were conducted the "furthest on circle" would just expand continuously until none of the 500 simulated targets were left in the area.

Updating for negative information takes into account the information embedded in an unsuccessful search over a period of time. During a certain time interval, some simulated targets are within the range of detection of some searcher. If the real target is not detected, then those simulated tracks become less likely and those further apart from the searcher become more likely.

Updating for positive information takes place when a

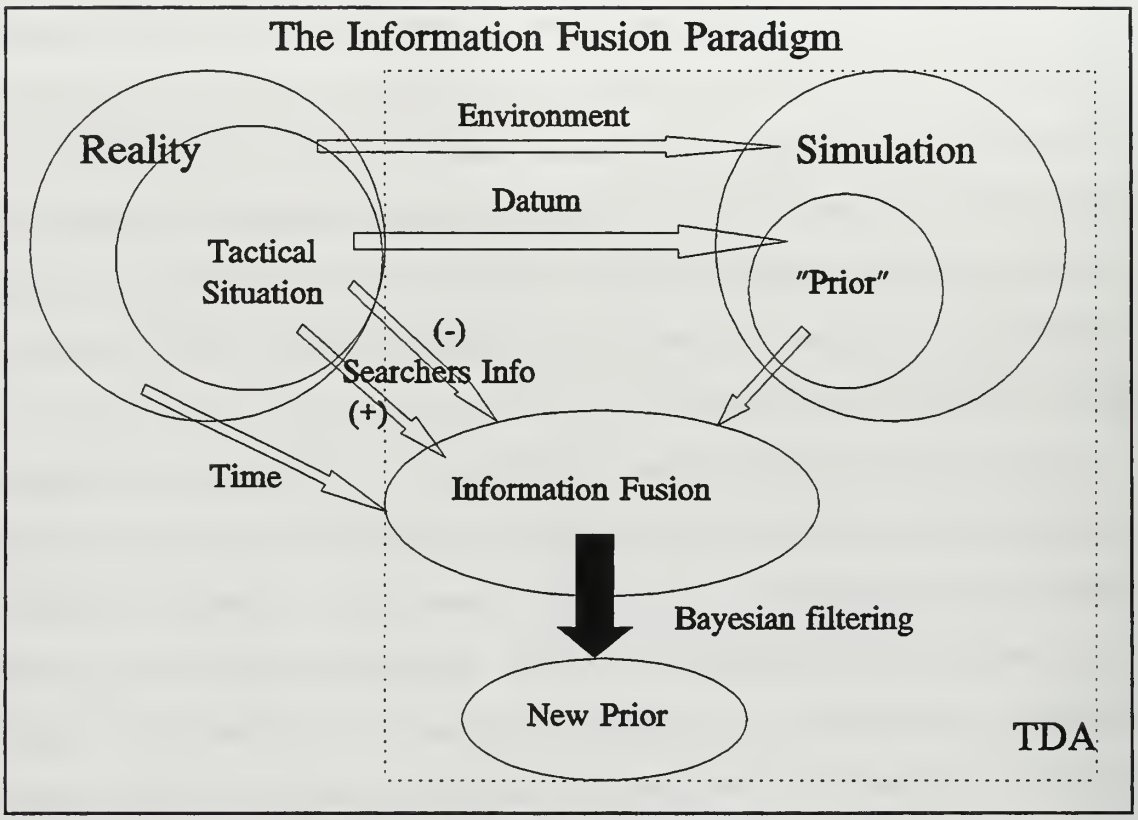

Figure 1.2. The Information Fusion Paradigm

searcher gains a contact. Those simulated targets within a certain distance from where the contact is produced become more likely than others further away. The probability map, the posterior or "new prior," consists of a partition of the Local Area into a number of cells. Each cell has a probability value which results from adding up all the individual likelihoods or weights of all simulated tracks within the cell. RTCAS produces a visual representation of the prior by mapping cell probabilities to a color code.

This successive updating of the individual weights of the simulated targets (Bayes Theorem) and the cell probabilities

(Law of Total Probability) at a given time interval is called Bayesian Filtering (Stone et al.,1995,pp.6).

2. Detection Effort

In a deterministic detection model like the "cookie cutter" sensor (Washburn,1989), a target within a certain range is detected with probability one, while those outside are not detected at all. The uncertainties involved in the detection process, intensified in the case of a shallow water environment and a small target, prevent the use of a deterministic model to evaluate the detection effort. Instead,the Fixed Scan Model (Washburn, 1989,pp.4), is used. For each pair simulated target-searcher, the Sonar Equation has to be solved yielding a value of Signal Excess (Kinsler et al., 1976,pp. 411) that represents how much above the Detection Threshold the signal return from the target is. However, the Signal Excess is considered a random variable due to the uncertainties in the problem, so a detection probability is calculated representing the detection effort. For example, a simulated target very close to the searcher will have a large value of Signal Excess chiefly because of the small transmission losses, thus leading to a high detection probability.

The same detection model serves to quantitatively appraise the effort in an unsuccessful search over an area as well as the likelihood of a contact report. The estimate of such an effort is given by the nondetection probability in the

former and the detection probability in the latter.

3. The Reactive Target

Active search turns the one sided activity of passive search into a game of two opposed goals. Parameterizing the reaction of the target is a complicated task because it involves human decisions. Selecting a course, depth, and speed or releasing a decoy or a weapon are the result of a sophisticated mental process that takes place in the target. The information is integrated in time to provide a picture of the threat the target faces versus its own resources to accomplish a goal. Modeling such a problem is a task out of the scope of this thesis. However, there are some basic parameters such as course and speed that can be considered the drivers of the situation. RTCAS uses those basic parameters to construct a simple model of the behavior of a real target.

D. THESIS STRUCTURE

Chapter II develops the core models for the Reactive Target, the Bayesian Filtering and the Detection Effort.

Chapter III explains the implementation of the models.

Chapter IV verifies the correctness and robustness in the implementation and the significance of a Reactive Target Model as opposed to a Non Reactive one.

Chapter V summarizes future enhancements of the model and the implementation.

Chapter VI states the conclusions and recommendations.

II. THE ANALYTICAL MODEL

A. THE TARGET MODEL

Three basic tasks have to be accomplished in the target model: generation, updating and reaction.

1. Track Generation

The initial prior in RTCAS is provided by a single scenario given by a lost contact or a flaming datum, i.e., the cue about the position of the target given by a ship in flames as a consequence of a torpedo attack.

Other TDA models,(Wagner,1989,pp.II.3), allow the generation of multiscenario priors but those are associated with large scale models that deal mainly with passive Area Search where several initial positions and estimated movements are likely and should be taken into account. In the case of a flaming datum, the expected movement is to flee the area.

This initial prior is represented by a bundle of 500 simulated tracks that encompass the many possibilities for target initial position and movement.

A Datum can be defined by the following elements

- Datum identity,
- time,
- position,
- error,
- course and speed,

RTCAS will deal with a single datum, hence the identification of the datum is irrelevant. In a Local Area Search, in general, all time and space coordinates are considered relatives to that of the datum, i.e., the geographical location of the area and the real time are not part of problem.

The time scale starts with the searchers' arrival in the Local Area after a *delay time* τ_d. (Figure 2.1)

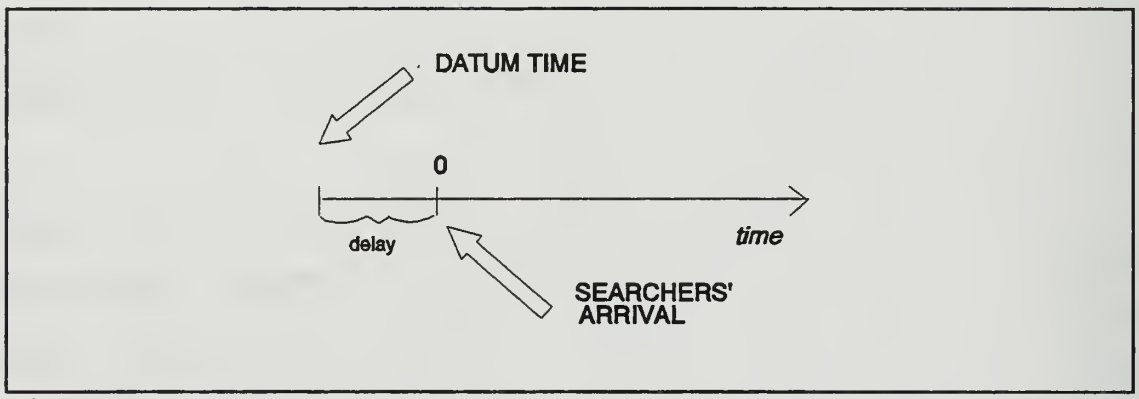

Figure 2.1. The time scale.

The positions of the tracks representing the datum are referred to the center of a grid that covers the Local Area where the search is conducted. The coordinates of any track *i* are $[x_{Ti}, y_{Ti}](t)$ at time *t*.

The error is taken into account when the tracks are generated and is assumed to be bivariate normal,

$$X_{Ti}(-\tau_d) \sim N(\mu_X, \sigma_X) \qquad (2.1)$$

$$Y_{Ti}(-\tau_d) \sim N(\mu_Y, \sigma_Y),$$

where the mean $\mu_X = \mu_Y$ are assumed to be equal to the center of the Local Area and the standard deviations $\sigma_X = \sigma_Y$ are input by the user. The notation X ~ N() means that X is a random variable normally distributed.

The magnitude of each target's velocity vector is given by a random variable uniformly distributed between a lower and an upper bound given by the user and does not change unless there is a reaction because of the presence of the searcher. (See Section 3).

The course is also a random variable uniformly distributed between user input limits. The purpose of this model is to resemble a situation where a preferred course of evasion exists,e.g., windward, towards its mission essential target, navigational hazard avoidance, etc. The situation where no preferences may be estimated about the target's intentions uses the same model including all possible courses between 0 and 360 degrees, thus leading to an omnidirectional fleeing datum.

The delay time τ is taken into account by updating the position of all tracks using each one's velocity vector to obtain the initial target distribution.

$$[x_{Ti}, y_{Ti}](0) = [x_{Ti}, x_{Ti}](-\tau_d) + \tau \bar{v}_i(-\tau_d) \qquad (2.2)$$

A typical target distribution at time zero can be seen in Figure 2.2 representing the scenario the searchers find when they arrive at the Local Area. To the left of the figure, the target is assumed to flee omnidirectionally. To the right is the case when the target's course is between 10 and 100 degrees.

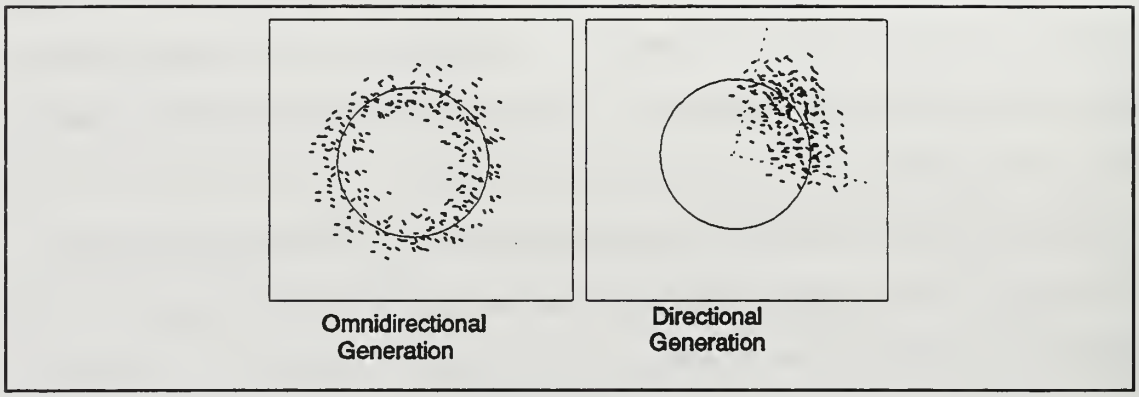

Figure 2.2. The initial target distribution.

2. Track Updating

At every time increment τ the position of all the targets as a function of time is updated using each track's velocity vector,

$$[x_{Ti}, y_{Ti}]\,(t+\tau) = [x_{Ti}, x_{Ti}] + \tau \overline{v}_i(t) \qquad (2.3)$$

The speed of the target remains unchanged unless there is a reaction.

3. Track Reaction

The reactive updating of the tracks consists of changes in the properties of each track as a consequence of the presence of the searchers. All reactions of a real target may be reflected in a variation of speed, course or depth. RTCAS updates the course only. The variations in depth are more likely to occur in deep water environments. Speed alterations, in general, are made within ranges that do not have a great impact in the overall geometry and kinetics of the problem.

The target has two goals, evade the datum and evade the searchers. In RTCAS the simulated targets evade the datum unless they have to evade a searcher because of its proximity. The assumption is made that the target knows the searcher's location. This is not too strong an assumption in active search. In general, the most common information the target knows about the searchers is the bearing. In the close distances normally implied in Local Area actions, the bearing varies rapidly thus allowing the target to perform an adequate passive motion analysis (Wagner,1989,pp. III-1). It is assumed that the target reacts when a searcher at a distance smaller than some threshold.

The simplest evasion maneuver is to adopt a course that generates a direction of relative motion (DRM) opposed to that of collision. Figure 2.3 , left depicts such a situation. Most of the time, this will be impossible for the target because its speed will be smaller than that of the searcher (Figure 2.3, right). A discussion of all possible courses of action available to the target at this point is out of the scope of this thesis. However, there are some constraints on what the target should do. It should not continue at the same course if the distance is decreasing, nor should it turn and end up with a relative bearing of ±90 degrees to the searcher, since this would increase dramatically the target strength in the sonar equation. A 180 degrees relative bearing should also be avoided since this may imply a reduction in the quality of

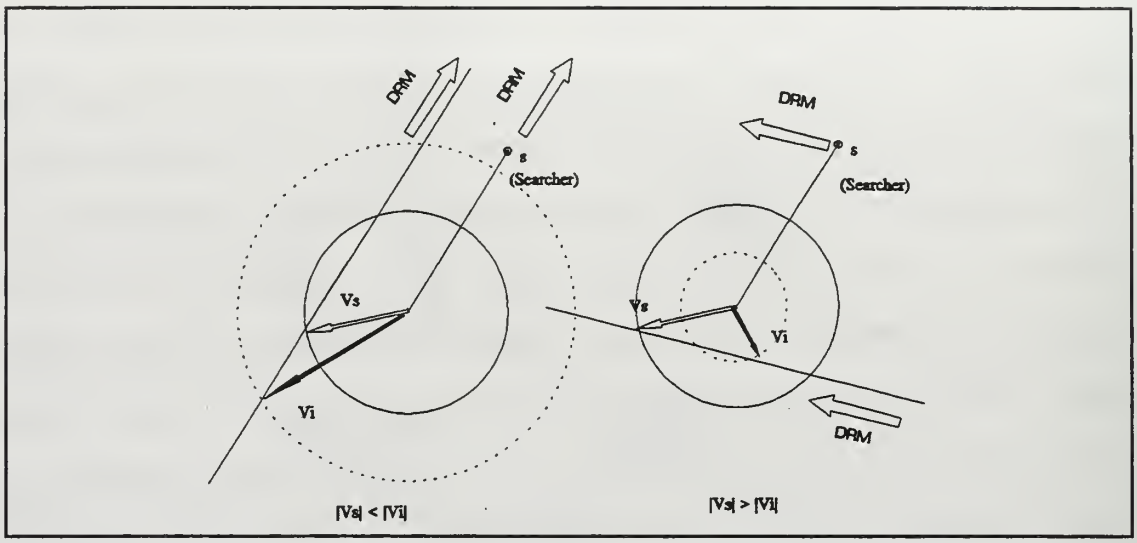

Figure 2.3. The evasion alternatives as a function of the velocities of the searcher V_s and the target V_i.

information obtained by the target because of the blind sector in the stern. For simplicity, it is assumed that target *i* turns a fixed angle $\Delta\theta \in (90°,180°)$ off the bearing B_{is} to the searcher. This new course for target *i* is,

$$\theta_i = B_{is} - \Delta\theta\, sign\left(\frac{dB_{is}}{dt}\right) \qquad (2.4)$$

where Db_{is}/dt is the bearing rate. If the bearing increases (decreases), the target turns to a course $\Delta\theta$ degrees measured counterclockwisely (clockwisely) from B_{is}. It is the tangential component of the searcher's speed relative to the target's velocity vector that determines the sign of the last term in (2.4) and it is calculated in RTCAS with the following formula,

$$sign\left(\frac{dB_{is}}{dt}\right) = sign(\sin(\Theta_s - B_{is})) \qquad (2.5)$$

where θ_s is the searcher's course.(Figure 2.4)

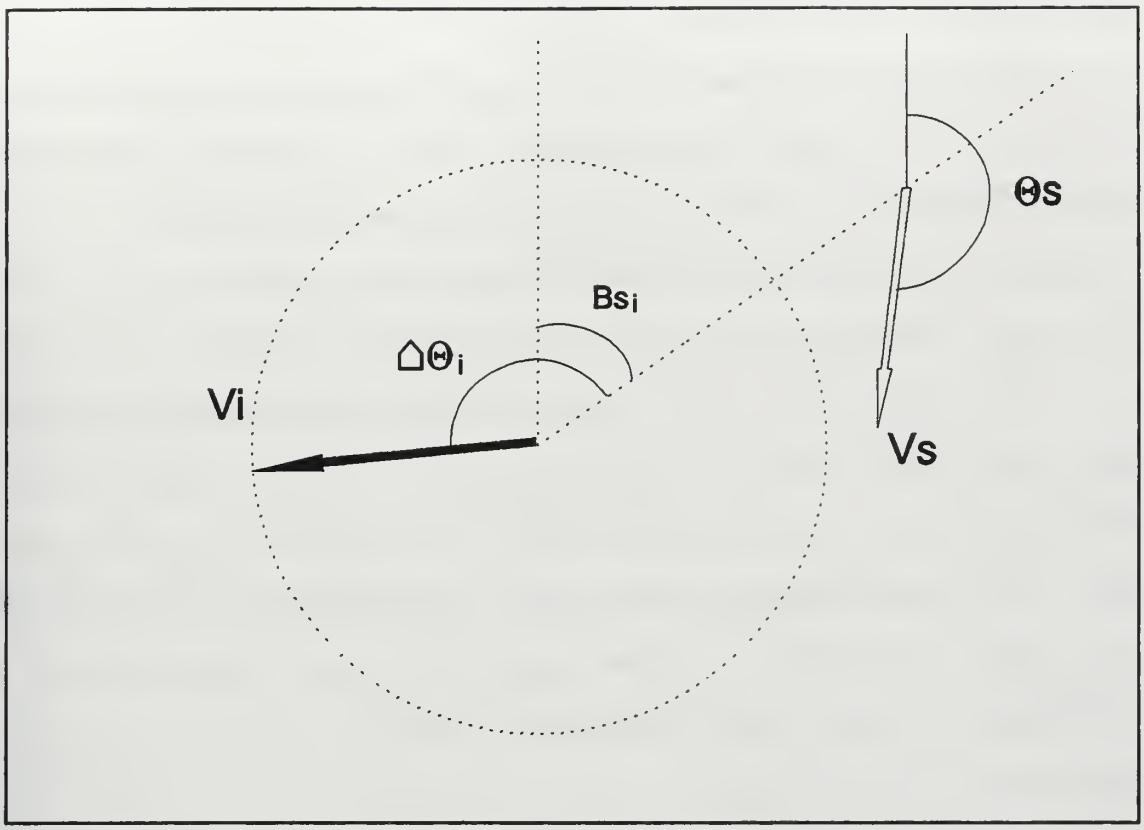

Figure 2.4. The evasive course parameters

When facing several searchers within the reaction range, the target decides to evade the closest.

B. THE BAYESIAN FILTERING MODEL

1. The Probability Map

The final purpose of the model is to provide a "probability map" representing the location of the target as an aid in conducting a Local Area Antisubmarine Active Search.

The search starts with a "flaming datum" (Cheong, 1988, pp.1), hence a relatively small time and space frame contains the action, e.g., 1 to 4 hours and an area with a radius from

10 to 20 nautical miles.

The probability map is a partition of the Local Area into a grid of *KxL* cells each of which has a probability value associated with it. We define the following events:

I :{ "*The target is within the Local Area*"}

G_{kl} :{ "*The target is in the grid cell k,l*"}

Then the probability map is the bivariate probability mass function of $P(G_{kl}\backslash I)$. The idea of a flaming datum as the starting point of the search makes the condition on the fact that the target in within the Local Area rather irrelevant at the start. However, as times goes by, the influence of I increases as the target may have already left the box where the search is conducted.

In order to build the probability map it is necessary to find a way to assign a value to $P(G_{kl}\backslash I)$.

This Bayesian analysis begins with a single scenario prior distribution of the "state of nature" (Washburn, 1995,pp.1) represented by N = 500 possible tracks. Intuitively, when conducting a Local Area search, one make some assumptions and evaluate a series of "what if" questions, reacting to the most evident and easy to maintain in one's mind as a chain of causes and effects. In this case N simulated tracks try to keep that chain alive for as long as possible.

The prior is generated by the Target Model as described in the previous section. The output of this model is, at this

stage, the stochastic location and movement of each one of the N simulated tracks according to the assumed prior. We define the event,

T_i: $\{$*"The real target follows track i", i=1..N*$\}$

then, if all tracks are equally likely to be the real target, assumption embedded in the idea of a single scenario, then the initial "weight" of each track is

$$P[T_i](0) = \frac{1}{N} \tag{2.6}$$

and the initial probability map can be depicted as

$$P[G_{kl}](t) = \sum_{i \in K_{kl}(t)} P[T_i](t) \tag{2.7}$$

where $K_{kl}(t) = \{i : [x_{Ti}, y_{Ti}](t) \subseteq G_{kl},\ t=0\}$.

Ideally, if there were no searchers and (2.7) was calculated as a function of time, it would convey almost the same information as the expanding furthest-on-circle (Cheong, 1988, pp.1). However, the purpose of RTCAS is to integrate the information from the real world provided by the assets conducting the search together with the information provided by the prior (Stone et al.,1995,pp.1). The search done up to a certain time t may be unsuccessful or result in one or more searchers obtaining a contact, which requires updating the prior for negative or positive information respectively (Wagner,1989,pp.II.5).

2. Updating for Negative Information.

The valuable information of an unsuccessful search during the interval τ is that the real target is less likely to be in the cells that had been searched as a result of the detection effort imposed to the tracks located in or near those cells. The value of each track weight will be decreased in accordance with that effort. The track weights will change from the prior values at time t to the posterior values at time $t+\tau$, hence changing the probability map of the location of the target.

Define the event,

$D_{s\tau}$:{ "*The target is detected by the searcher s during time interval* τ" }

then

$$cdp_{si\tau}(t) = P[D_{s\tau} \mid T_i](t) \qquad (2.8)$$

is the cumulative probability that the target was detected by the searcher s during the time interval τ starting at time t conditioned on that the real target was located at the position of simulated target i during that interval. This value is provided by the Detection Model to be developed in section C.

If $D'_{s\tau}$ is the complement of $D_{s\tau}$, then

$$P[D'_{s\tau} \mid T_i](t) = 1 - cdp_{si\tau}(t) \qquad (2.9)$$

is the nondetection probability of searcher s over target i during interval τ starting at time t. Consequently, the probability that the target is following track i and is not

detected by searcher s during the interval τ is,

$$P[T_i \cap D'_{s\tau}](t) = P[D'_{s\tau} \mid T_i](t)\, P[T_i](t) \qquad (2.10)$$

Using Bayes Theorem, the Negative Information Updating of the prior is

$$P[T_i](t+\tau) = \frac{P[D'_{s\tau} \mid T_i]\, P[T_i](t)}{\sum_{i \in N} P[T_i \cap D'_{s\tau}](t)} \qquad (2.11)$$

Note the convention that $P[T_i](t+\tau)$ is conditioned on all information available up to time $t+\tau$.

The denominator in (2.11) represents the unconditional probability $P[D'_{s\tau}](t)$ obtained using the Total Probability Theorem. Furthermore it is the normalizing factor that will allow the new prior to be a "proper" probability mass function i.e. $\Sigma_{i \in N}\, P[T_i](t+\tau) = 1$.

An iterative application of this process permits the calculation of the new prior for any time. The update 2.11 must be applied once per time interval for each searcher that does not detect the target.

3. Updating for Positive Information

The probability map should be updated for Positive Information when a searcher reports a contact. It may be contended that once a contact is obtained, the relative value of the information provided by the simulation of tracks is somehow negligible with respect to the real world information implied in the contact report. However, in a shallow water acoustic environment where many elements of the "clutter"

space,e.g., a whale, a rock, an eddies, may be considered a contact, the fusion of information of both sources remains in place as a means to smoothing the reactions and providing continuity whenever the contact is confirmed not to be the desired target.

Define the following events

$C_{sc\tau}$:{ *"The searcher s reports a contact at the position c during the interval τ", s=1..S, c=1..C*}

where *S* is the total number of searchers and *C* is the total number of reported contacts among all searchers, and,

$E_{c\tau}$:{ *"The real target is located at the position of the contact report c during the time interval τ"*}

then,

$$C_{sc\tau}=D_{s\tau}\bigcap E_c \qquad \textbf{(2.12)}$$

where, as before, $D_{s\tau}$ is the event that searcher *s* detects the target. Thus,

$$P[C_{sc\tau}\mid T_i](t)=P[D_{s\tau}\mid T_i](t)\,P[E_c\mid D_{s\tau},T_i](t) \qquad \textbf{(2.13)}$$

The first factor in 2.13 is the detection effort alloted by searcher *s* at the position of simulated track *i* during period *τ*,

$$P[D_{s\tau}\mid E_c](t)=cdp_{si\tau}(t) \qquad \textbf{(2.14)}$$

Assuming that location errors are circularly normally distributed the second factor is

$$P[E_c \mid D_{s\tau}, T_i] = ke^{\frac{-r_{ic\tau}^2}{2\sigma_{si}^2}} \qquad (2.15)$$

where $r_{ic\tau}$ is the distance between the position of the contact report c and track i during interval τ (Figure 2.5), k is a constant independent of c and i, and σ_{si} is the standard deviation of reporting errors made by searcher s for a target following track i during interval τ. If Ω_s is the beamwidth of

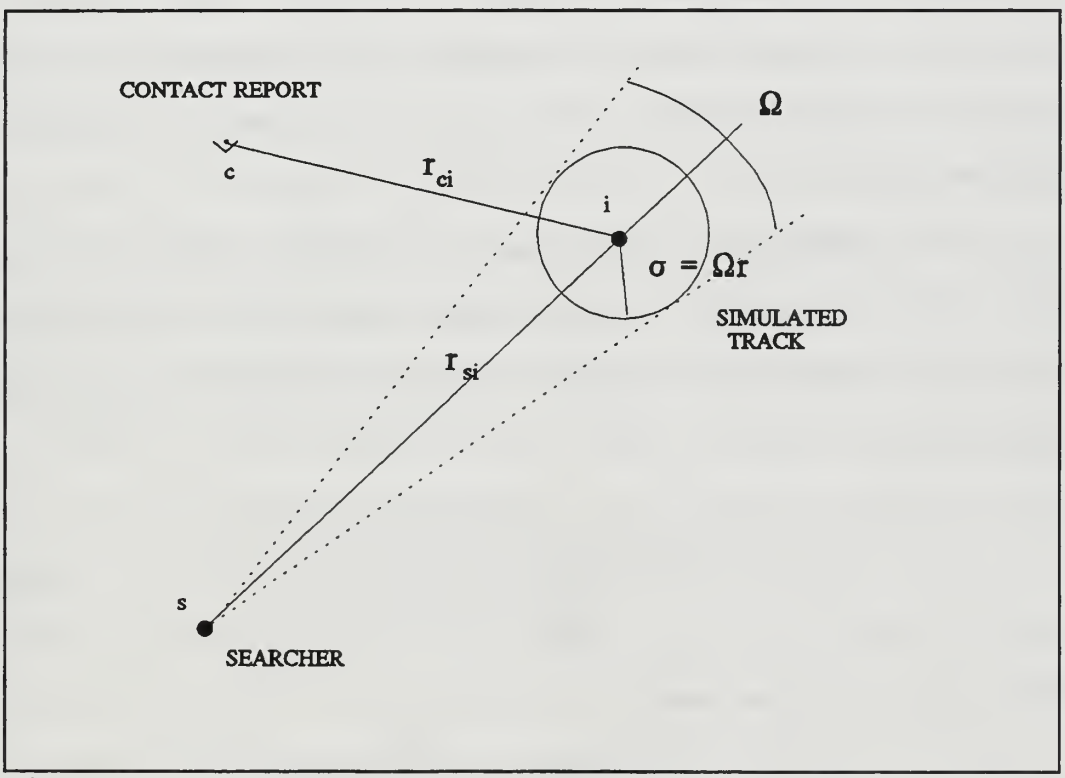

Figure 2.5. The Error in the contact report.

the sensor and r_{si} is the distance between searcher s and tracki, then RTCAS takes σ_{si} to be $\Omega\, r_{si}/2$. Thus 2.15 becomes,

$$P[E_c \mid D_{s\tau}, T_i] = e^{\frac{-2r_{ic\tau}^2}{\Omega_s^2 r_{si}^2}} \tag{2.16}$$

where the constant k was dropped because it cancels when Bayes Theorem is applied. Substituting 2.14 and 2.16 into 2.13,

$$P[C_{sc\tau} \mid T_i]\,(t) = cdp_{si\tau}(t)\, e^{-\frac{r_{ic}(t)^2}{\Omega_s^2 r_{si}(t)^2}} \tag{2.17}$$

represents the probability that searcher s reports a contact at position c given that the target follows track i during the interval τ. Equation 2.17 embodies the mathematical core of fusing information between the real world contact report and the simulation.

The probability that the real target follows track i given that the searcher s reported a contact at the position of c during period τ applying Bayes Theorem is (Washburn, 1995,pp.6),

$$P[T_i \mid C_{sc\tau}](t) = \frac{P[C_{sc\tau} \mid T_i]\, P[T_i]\,(t)}{\sum_{i=1}^{N} P[C_{sc\tau} \mid T_i]\, P[T_i]\,(t)} \tag{2.18}$$

which using 2.17 becomes

$$P[T_i]^{+}(t+\tau) = \frac{cdp_{si\tau} e^{-\frac{r_{ic}^2}{\Omega_s^2 r_{si}^2}} P[T_i]\,(t)}{\sum_{i=1}^{N} cdp_{si\tau} e^{-\frac{r_{ic}^2}{\Omega_s^2 r_{si}^2}} P[T_i]\,(t)} \tag{2.19}$$

where the + superscript implies that the contact is actually a true contact on the target. In practice, updates for positive information must allow for the possibility of false alarms, so the formula actually used by RTCAS is,

$$P[T_i](t+\tau) = P[T_i]^+(t+\tau)\,Q + P[T_i](t)\,(1-Q) \qquad (2.20)$$

where Q is the "credibility" (Washburn,1995,pp.2) of the contact report. When $Q=1$, 2.20 is the same as 2.19, but when $Q=0$, 2.20 is just the prior $P[T_i](t)$.

It might be contended that the posterior alternate to $P[T_i]^+(t+\tau)$ should include the negative information implied in the unsuccesful search that a failed contact report conveys. However, it is very plausible and commonly seen in the practice that a searcher investigating a contact will focus onto it, not only because of a natural tendency in the distribution of the attention of the sonarman, but also because investigating a contact requires more often than not using modes or techniques in the sensors that prevent the operator from an adequate scanning of the rest of the picture even without intention of doing so. Examples of those modes or techniques are doppler and image analysis, noise reduction by windowing the signal processing in the target, angular scanning reduction spotting to concentrate acoustic intensity on the target, etc. As a consequence and for computational convenience, RTCAS considers an absolute focusing on the contact, i.e.,the individual track probability alternative to

the one updated for positive information is the value at the beginning of the scan period as in equation 2.20.

4. The undetected probability mass

Although each probability map is the best estimate of the location of the target, it says nothing about how much of the original probability mass remains undetected. A measure of the effect of the search is necessary since the searcher needs to know how much probability mass is still undetected in order to be able to decide when to terminate the search.

The denominator in (2.11) is the probability that the real target is following one of the 500 simulated tracks and was not detected, hence is the undetected probability mass during the interval τ at time $t+\tau$:

$$S_\tau(t) = \sum_{i \in N} P[T_i \bigcap D'_{s\tau}] \qquad \textbf{(2.21)}$$

for instance, at time $t=0$, before any search is conducted $S(t)=1$. At each updating step the searching will reduce the original probability mass such that at any time t, the Cumulative Undetected Probability Mass is

$$U(t) = U(t-\tau)\, S_\tau(t) \qquad \textbf{(2.22)}$$

This recursive definition allows the calculation of its value at every time step in order to produce the desired assessment.

C. THE DETECTION MODEL

Several possibilities were taken into account in select-

ing the detection model. The idea is to model active search. The Fixed Scan Model seems to be naturally applicable since (Washburn,1994, pp. 3):

- Each pulse can be thought of as an independent look,
- Detection in the absence of a pulse is impossible,
- There exists a pulse rate which can be associated with the pulse repetition frequency of a sonar.

The first statement is seemingly too strong an assumption but in RTCAS only the conditional independence of the pulses given the target track, is required. The validity of the assumption depends on the rapidity of fluctuations in the sonar, equation terms relative to the time between active pulses.

The pulse repetition frequency (prf) is normally associated with the maximum range at which it is desired to attempt a detection. Given a typical value of maximum scale range in active sonar of about 18000m, the round trip of a pulse takes 24 seconds at a sound speed of 1500m/s, thus yielding a prf of approximately 2 per minute. In conventional active sonar the frequency is normally greater than 4 kilohertz. The strongest time variability in the propagation is given by the time change in the refraction patterns which in turn depend on the depth-velocity profile. In shallow waters, the fastest change in the upper layer of the velocity profile can be regarded as semi-diurnal as a consequence of the heat interchanges

provoked by the daily cycles of the sun. Nonetheless, the effect of cloud cover, rain, wind or other many factors may produce noticeable changes in the propagation conditions. Furthermore, there are some changes in the geometry of the detection problem that may cause variations in the time frame of interest. For instance, given a frequency of 4.5 kilohertz, the wavelength will be 0.3 meters. In shallow waters, it is very likely that the receiver will sense a multipath scattering from the target. If the difference between two paths is a multiple of 0.15 meters, there will be destructive interference thus leading to an eventual lack of detection. Although a deterministic geometric model may account for each situation regarding the relative positions of target and searcher with respect to the bottom, the surface and the layer, the outcome of such a model may be regarded as randomly distributed with respect to space and time. Other phenomena may concur in order to increase the randomness in the signal at the receiver between different pulses, for instance, the presence of the bubbles due to the wake of other ships or the target, changes in the angle of incidence of the pulse at the target, the destructive interference with the signals from other platforms, the macroscopic biological activity, etc. All of these sources of randomness are typical of the shallow water active signal propagation problem.

The assumption in RTCAS is that **each** pulse is conditionally independent of the others. Of course the computational

difficulty in considering nonindependence had to be taken into account and favored the chosen option.

1. The Fixed Scan Model

This thesis applies the Fix Scan Model as a paradigm of the use of hull mounted and dipping active sonars and sono-bouys in searching for a conventional submarine. However, it is applicable to other types of sensors like radar, lidar, photo electronic devices, etc. as well as other types of targets like surface ships, life rafts, land vehicles, etc.

The sensor is assumed to emit pulses at a repetition frequency, *prf*, given by

$$prf=\frac{2C}{M} \qquad (2.23)$$

where C is the speed of the sound in the water and M is the maximum unambiguous range, assumed always to be equal to the maximum range scale selected in the device.

The probability of detection will be calculated only for those targets within the maximum range scale of any searcher at the end of every *updating time interval* τ. The value of τ must satisfy a compromise between the need of an adequate "sampling" of the search process and the computational difficulty of too many calculations in a given time frame.

As an example, for a maximum range scale of 18000m, the interval between pulses will be 24 seconds, which may be approximated as 30 seconds. Hence, if the updating interval is

τ = 5 minutes, 10 independent scans are going to be made with the same sonar operating characteristics, i.e., same pulse type, prf, frequency, etc. Selecting 5 minutes as the updating interval is convenient from the point of view of the computational effort as well as tactically reasonable; e.g., the rate of turn of a conventional submarine at 6 knots is such that it would take about 4 minutes to complete a 180° turn.

During one updating interval τ, both the searcher and the target are assumed to remain stationary at a point midway between their respective locations at the beginning and at the end of τ (Figure 2.6), for purposes of detection probability calculations.

Assuming the above mentioned independence between each scan, (Washburn,1995,pp.3), the probability that at least one of the pulses k from searcher s detects the target given that it follows track i, at time t, is,

$$cdp_{\tau si}(t) = 1 - \prod_{k=1}^{T} (1 - p_{ksi}(t)) \qquad (2.24)$$

where T, the total number of pulses during the updating interval τ, may be calculated as,

$$T = \tau * prf \qquad (2.25)$$

The detection probability of each pulse k is given by,

$$p_{ksi}(t) = \Phi\left(\frac{SE_{si}(t)}{\sigma_N}\right) \qquad (2.26)$$

where Φ is the Standard Normal Cumulative Density Function and

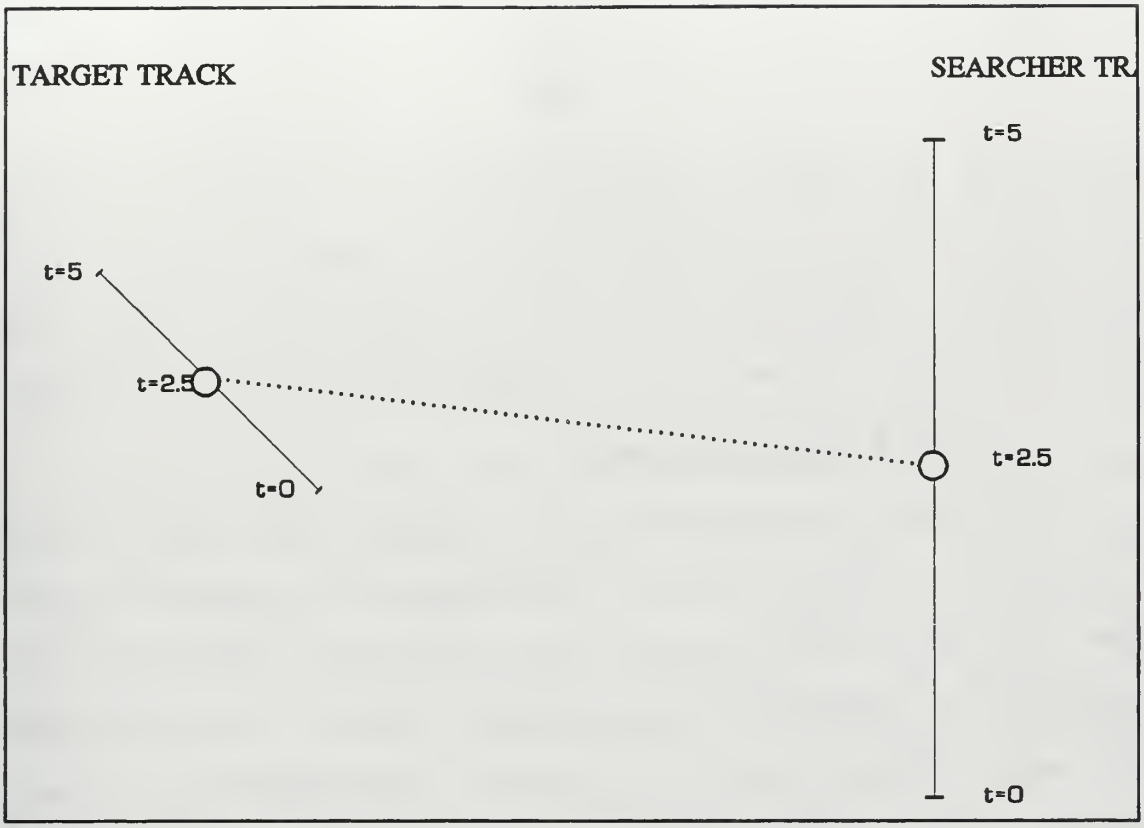

Figure 2.6. The kinematics of the detection process

$SE_{si}(t)$, calculated by the Propagation Model to be described in the next section, is the Signal Excess from target i at the receiver in searcher s at time t, and σ_N is the standard deviation in the calculation of SE_{si} due to

- the randomness in the detection process, and
- the cumulative error or uncertainties in the calculation of each one of the terms in the sonar equation.

The value of σ_N can be estimated either by experience or by using a property of the variance of the sum of random variables (Devore,1991,pp.102),

$$\sigma_N = \sqrt{\sum_{k=1}^{S} \sigma_k^2} \qquad (2.27)$$

where σ_k is the error in the calculation or estimation of each k of the S terms in the sonar equation. Wagner (1989,pp. II-28) suggests values of 6-9 dB for σ_N, RTCAS uses 15 dB since it is intended for use with relatively poor sonar forecasts.

2. The Propagation Model

This model corresponds to the propagation of active sonar signals in a shallow water environment. The values of all terms in the sonar equation will be either assumed to be constants or approximated by simple models. It is the user that will finally input the values corresponding to more accurate calculations.

For each pair searcher s and target i at time t the signal excess is be calculated using the active sonar equation, (Kinsler et al. 1982) limited either by noise,

$$SEN_{si} = SL_s - DNL_s(t) - DT_s + DI_s - 2TL_{si}(t) + TS_{si}(t) \qquad (2.28)$$

or by reverberation,

$$SER_{si}(t) = TS_{si}(t) - DT_s - SS_s + TLG_{si}(t) - 10\log_{10}\left(\frac{c\tau\Omega}{2}\right) \qquad (2.29)$$

All levels in the formulae above are in decibels with respect to one microPascal. The Source Level SL, the Detection Threshold DT and the Directivity Index DI are considered

constant for each searcher s. The rest of the terms are variables calculated by RTCAS and are developed in the following sections. DNL_s is the Detected Noise Level for searcher s, $TL_{si}(t)$ is the Transmission Loss for each pair (s,i) at time t, $TS_{si}(t)$ is the Target Strength of target i with respect to searcher s, SS is the Scattering Strength for the sonar in searcher s, $TLG_{si}(t)$ is the transmission loss due to geometrical spreading and the last term in (2.10) is the area of the reverberating surface within the beam of the transmitted pulse corresponding to the speed of the sound c, the pulse length τ and the horizontal angle of the sonar beam Ω (Kinsler et al.,1892). The smaller of (2.28) and (2.29) is to be selected and used in calculating (2.26).

In active search in shallow waters,the sonar equation is, in general, more likely to be limited by reverberation rather than by noise. However, in some cases the effect of the noise will prevail at distances within the maximum range scale since the echo level and the reverberation level decrease with range but the noise level remains the same. A typical behavior can be seen in Figure 2.7, where up to 7500 meters the noise restricted Signal Excess is higher than the reverberation restricted one.

The following sections discuss the terms in the sonar equations (2.9) and (2.10) that are not considered a constant.

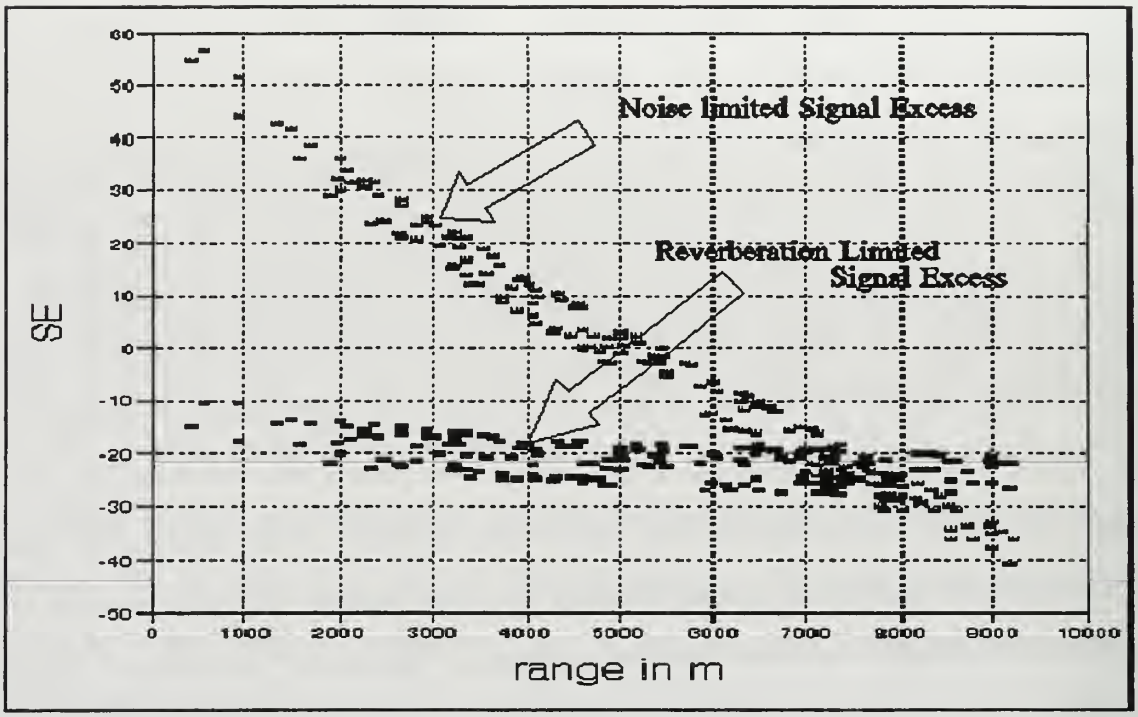

Figure 2.7. Noise and Reverberation Limited detection.

3. Target Strength

Target strength is defined as the echo returned by the target and hence is a factor when dealing with active sonar. The corresponding term in the sonar equation is:

$$TS_{si}=10\log_{10}\left(\frac{I_i}{I_{r0}}\right), r=1 \qquad (2.30)$$

where I_r is the intensity of return at 1 yd from the target and I_i is the incident intensity.

The intensity ratio I_r/I_i is a function of distance between the target and the source (Urick, 1975,pp.284), the shape of the target (Urick,1975,pp.274), the pulse length (Urick,1975,pp.285), the frequency of the incident signal

(Urick,1975,pp.283), and the geometry of the reflection (Urick,1975,pp.281). The influence of all those factors except the last can be considered constant.

In the case of a monostatic sonar the direction of incidence is the same as the direction in which the signal is received at the transducer except for small variations due to the relative motion target-searcher during the time the pulse travels. The target strength is a function of the angle between the heading of the submarine and the angle of incidence of the signal, which is called *aspect angle*. If lateral symmetry is assumed, then the aspect angle can be considered from 0 to $\pm \pi$, with 0 being at the bow. The great variation of *TS* as a function of the aspect angle requires the aspect angle to be considered. The Target Strength will, in general, vary from its maximum value at broadside to a minimum at the bow and stern.

Let **r** be the distance target-searcher and **v** the target speed, then,

$$\cos(\theta) = \hat{r} \cdot \hat{v} \qquad (2.31)$$

indicates the dot product between the corresponding unit vectors and θ is the smallest angle between them, in this case the aspect angle (Figure 2.8).

The typical values of *TS* vary from about 25 dB at beam aspect to about 15 dB at bow and stern aspects (Urick, 1975 p.282). The following equation is an approximation adopted as

a simplified model of the variations between those extremes,

$$TS(\theta) = TS_o + \Delta TS * (1 - \cos^2(\theta)) \qquad (2.32)$$

where TS_o is the value of Target Strength at bow and stern, ΔTS is the typical increment to reach the value at beam aspect and $(1-\cos^2(\theta))$ models the variation as a function of the aspect angle θ. Substituting (2.12) into (2.13) yields,

$$TS_{si}(t) = TS_o + \Delta TS(1 - (\hat{r}_{si}(t) \cdot \hat{v}_i(t))^2) \qquad (2.33)$$

where $r_{si}(t)$ is the instantaneous distance between searcher s and target i, and $v_i(t)$ is the speed of target i at time t. RTCAS uses a square of the cosine as a model because of

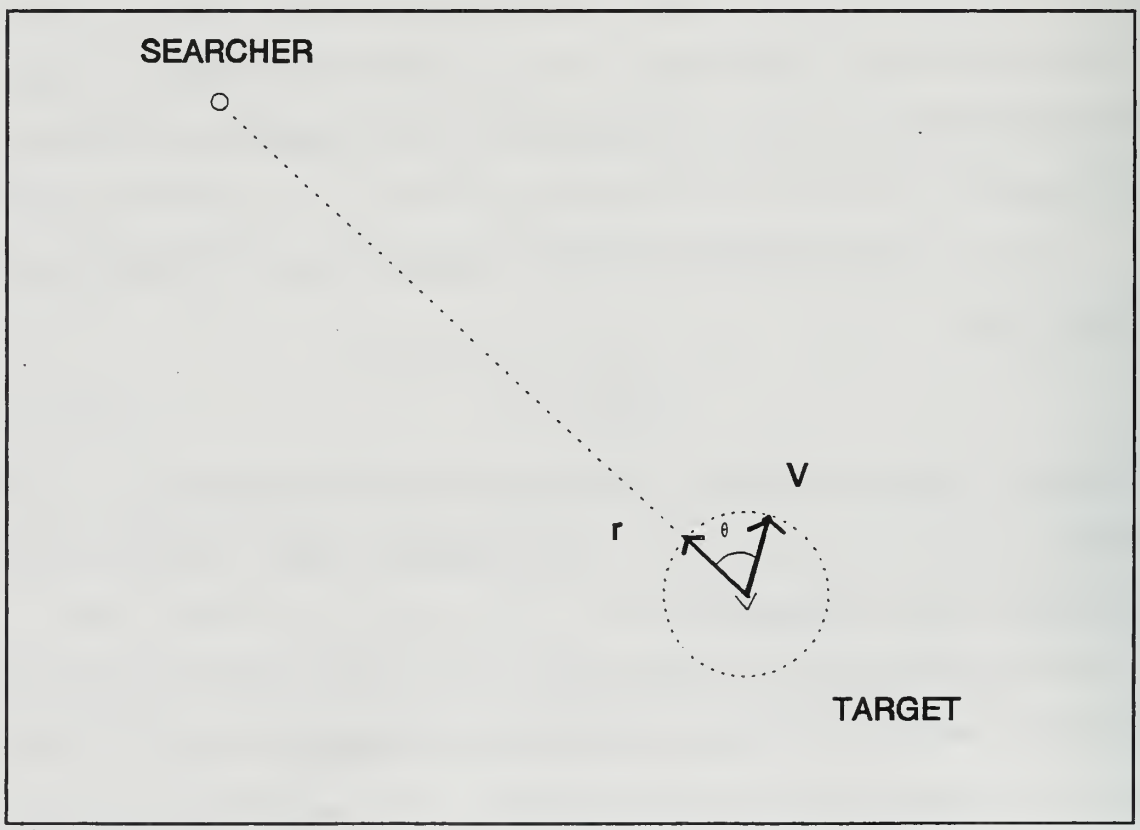

Figure 2.8. Vector representation of the target strength

computational convenience. Note in 2.33 that the trigonometric function in 2.32 is replaced by a dot product, which can be calculated using the components of the corresponding unit vectors instead of the trigonometric function.

4. Transmission Loss

Transmission loss involves the combined effect of geometrical spreading and absorption.

For simplicity, assuming that the distance $r_{si}(t)$ is much larger than the depth H, geometrical spreading is calculated with the following isospeed shallow-water channel perfectly rigid bottom expression (Kinsler,1984,pp.429),

$$TLG_{si}(t) = 10\log[r_{si}(t) + \frac{H}{\pi})] \qquad (2.34)$$

where the last term is the correction for the transition range.

The total Transmission Loss is

$$TL_{si}(t) = TLG_{si}(t) + ar_{si}(t) \qquad (2.35)$$

where a, the absorption coefficient, is calculated with

$$a = \frac{8\times10^{-5}}{\frac{0.7}{f^2}+1} + \frac{0.04}{\frac{6000}{f^2}+1} + 4\times10^{-7}f^2 \qquad (2.36)$$

from Kinsler et al.(1976,pp.398) where f is the frequency in Hertz at which each sonar operates.

5. Noise Level

The Noise Level accounts for the background ambient noise *(an)* and the platform self noise *(sn)*,

$$NL=20\log_{10}\left[\frac{an+sn}{1\mu Pa}\right] \qquad (2.37)$$

Instead of calculating the intensities *an* and *sn* their corresponding levels *ANL* and *SNL* in decibels are interpolated from observational data.

The Ambient Noise Level is calculated by means of a third order polynomial fit on windspeed and a linear fit on the logarithm of frequency of observational data obtained from curves representing the noise spectra for some coastal locations in Figure 7.7. in Urick (1975,pp.191),

$$ANL=46.12+2.22w-0.041w^2-17.01[\log_{10}(f)-3] \qquad (2.38)$$

where w is the wind speed in m/s and f is the frequency in hertz. This approximation is centered around f = 1000 Hz. Figures 2.9 and 2.10 compare Equation 2.38 to the data from the above mentioned reference.

The Self Noise Level is calculated as a function of frequency and the searcher's speed using figure 11.11 in Urick(1975, pp.340), which shows equivalent isotropic self-noise levels at 25 khz on a number of World War II American and British destroyers. A polynomial fit of this data, including an extrapolation term for other frequencies assuming a slope of -6dB per octave, yields,

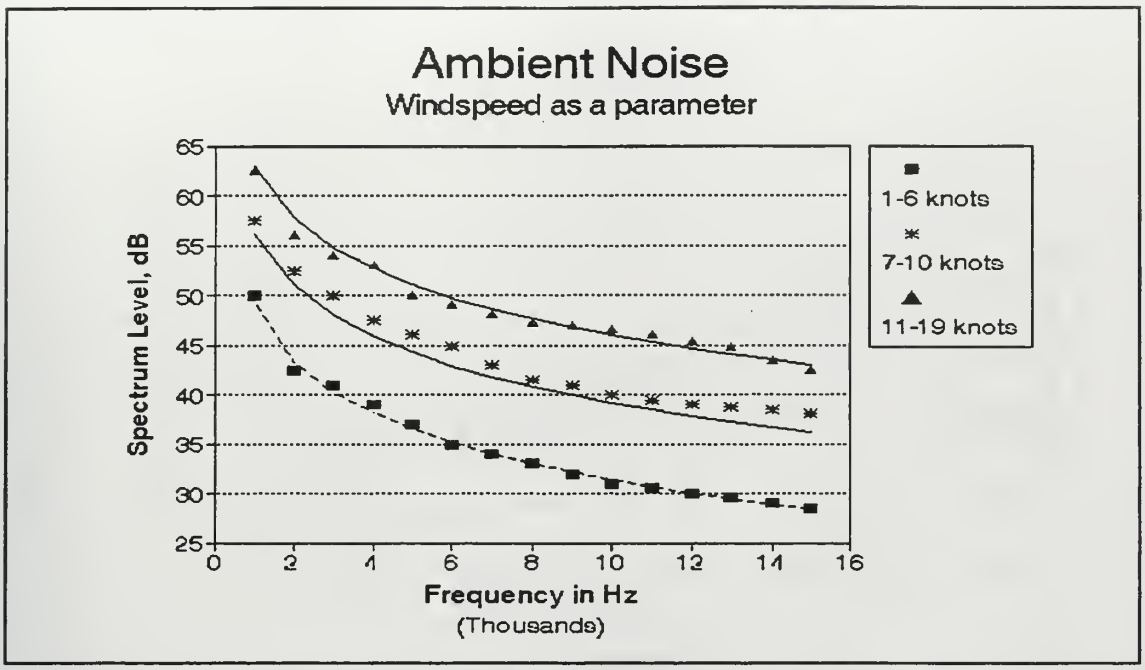

Figure 2.9. The Ambient Noise as a function of frequency and the windspeed as a parameter between 1 and 16 knots.

$$SNL = 23 + 2.02 v_s(t) + 20 \log\left(\frac{25}{f}\right) \qquad (2.39)$$

where v_s is the searcher's speed in knots and f is the frequency in kilohertz. Figure 2.11 compares this equation with the corresponding data. SNL and ANL are power-summed to obtain NL.

6. Scattering Strength

According to Urick (1975,p.253), when downward refraction occurs, i.e., negative sound speed profile, bottom back scattering is dominant and the reverberation level can be calculated using the bottom scattering strength. Conversely, when upward refraction occurs, i.e., positive sound speed profile as in isothermal water, the surface backscattering

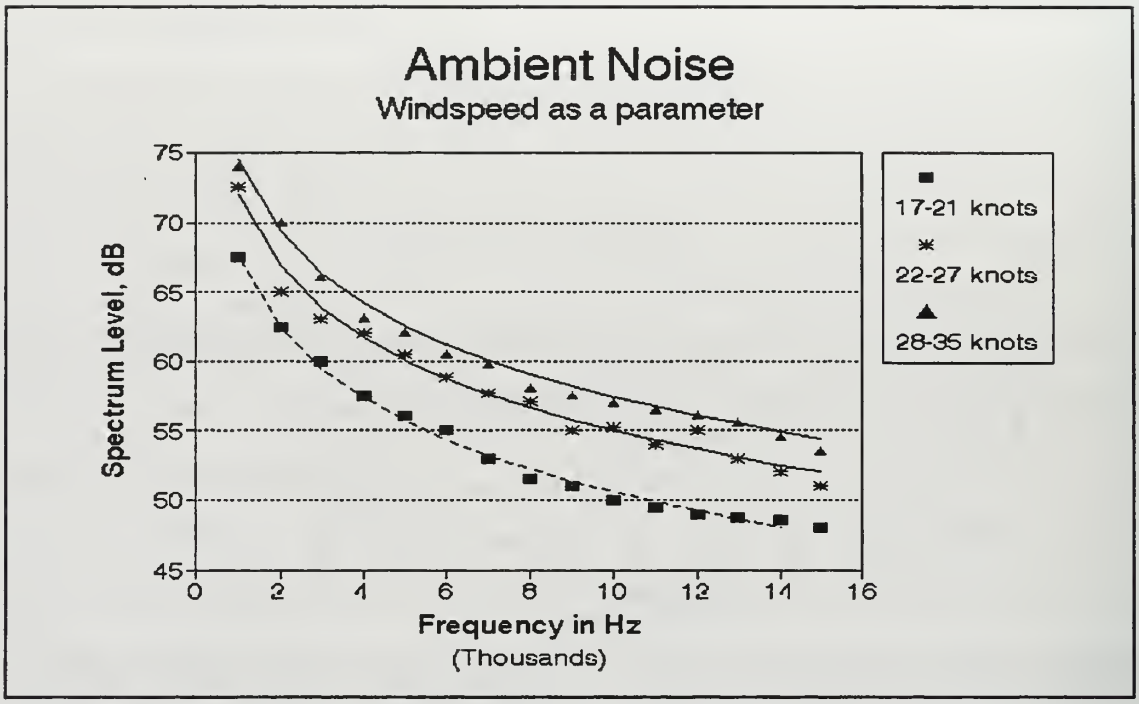

Figure 2.10. The Ambient Noise as a function of frequency and windspeed as a parameter between 17 and 35 knots.

becomes most important. This model follows that rule.

Following Urick (1975,pp.241),the Surface Scattering strength is calculated with,

$$SSL=10\log_{10}(f\times h\times\sin(\theta))^{0.99}-45.3 \qquad \textbf{(2.40)}$$

where f is the frequency in hertz, θ is the grazing angle in degrees and h = 0.0026 $W^{5/2}$ where W is the windspeed in meters per second. In a shallow water environment,the backscattering of the sonar signal will occur at all possible grazing angles due to the multiple interactions between the surface and the bottom. Due to the analytical and computational difficulty in accounting for those angles, θ was chosen to be 35°, an average between the typical limit angles in Figure 8.21 in Urick(1975, pp. 240).

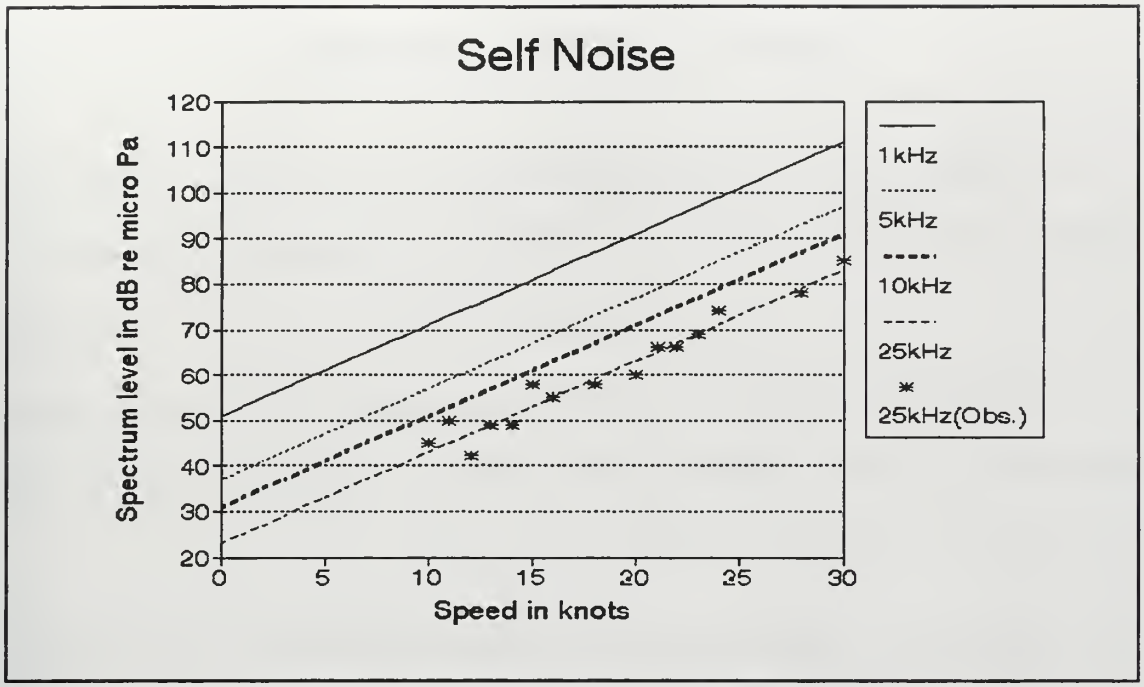

Figure 2.11. Self Noise data for 25kHz, its polynomial fit and the extrapolations for other frequencies using a slope of -6dB per octave.

In the calculation of bottom strength, Urick (1970,pp. 396) found that the spectrum slope in shallow waters was similar to measurements made in deep waters, therefore figure 8.30 in Urick (1975, pp. 248) could be used as an approximation for shallow waters. Depending on three generic types of bottom, the backscattering *Bs* is calculated using a third order polynomial fit of the data in figure 8.30 as a function of frequency as follows

- TYPE I : Plain bottom, little or no roughness, high angular and frequency variations.

$$BS=0.012f^3+0.283f^2+0.0062f-28.153 \qquad (2.41)$$

- TYPE II: Intermediate between a very rough, rocky bottom and a Type I bottom

$$BS=0.076f^3-0.233f^2+2.8706f-40.41 \quad (2.42)$$

- TYPE III: Heavily dissected bottom, with underwater ridges, very rough. Almost no angular or frequency dependence,

$$B_s=-18 \quad (2.43)$$

Figures 2.12 and 2.13 show the data and the corresponding polynomial fit for bottom types I and II. The user must decide which type of bottom to use.

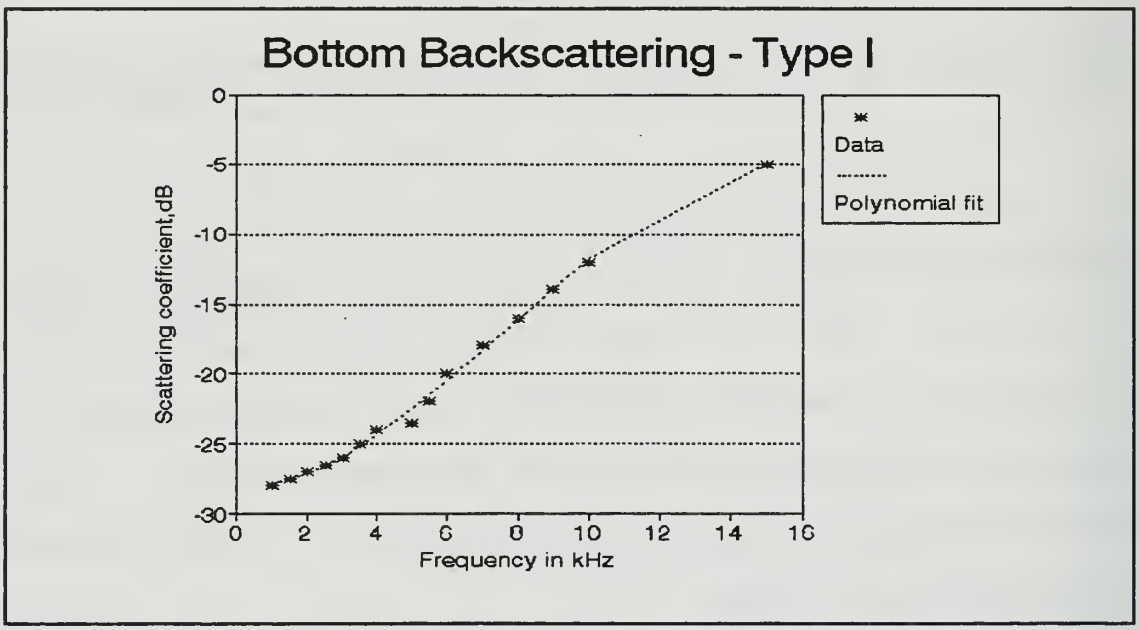

Figure 2.12. The bottom backscattering for plain surfaces.

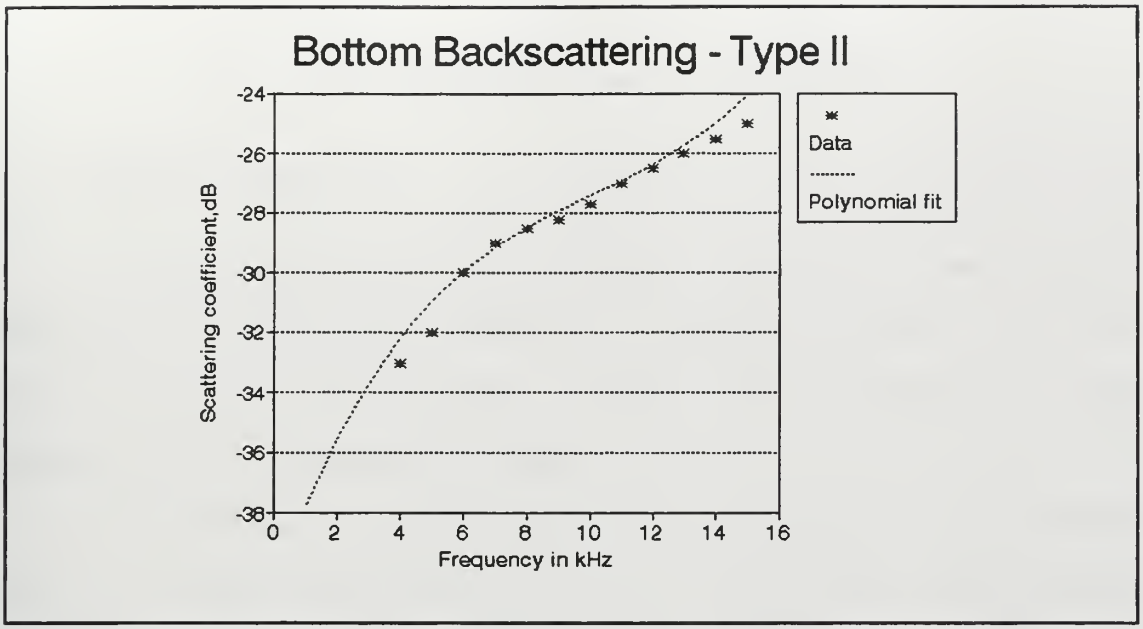

Figure 2.13. The bottom backscattering for intermediate surfaces.

III. THE IMPLEMENTATION OF THE MODEL

A. PROGRAM STRUCTURE

The analytical model described in Chapter II was implemented in a structural procedural program written in Turbo Pascal 7.0 for DOS (Borland, 1991). The structure of the program can be observed in Figure 3.1.

The program consists of a main program named RTCAS (Reactive Target Computer Assisted Search) and seven units namely, InterManager, ScoutManager, TargtManager,Detection Manager, CellsManager and GraphManager.

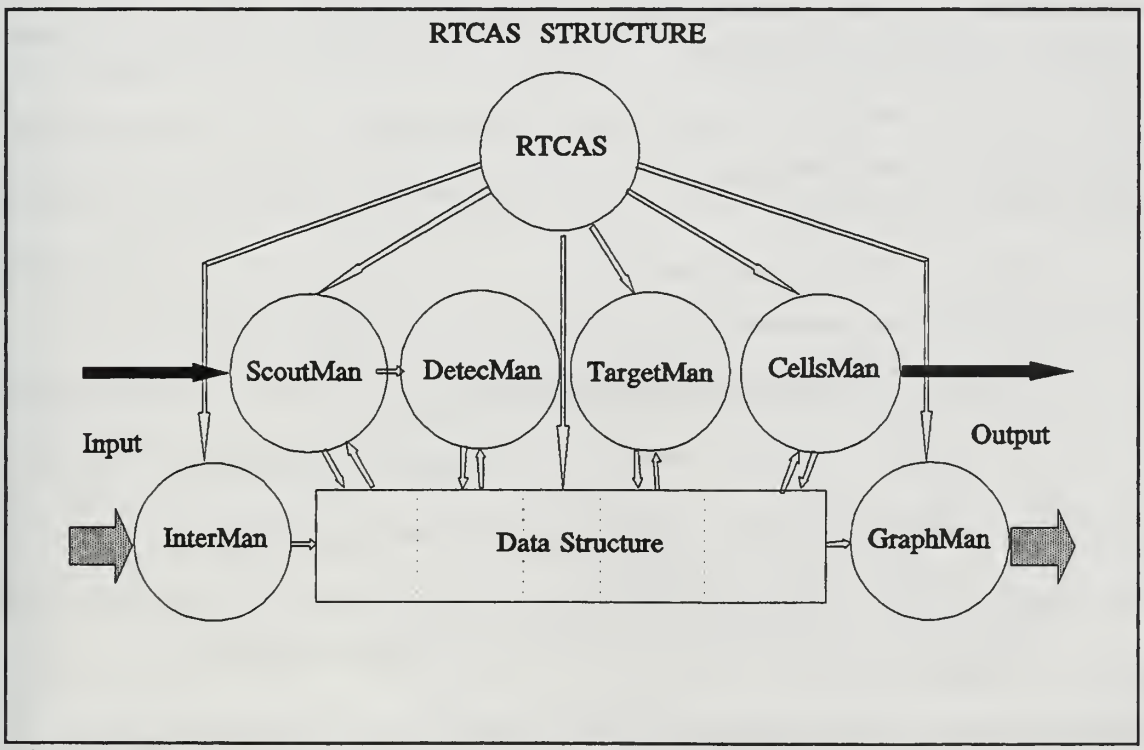

Figure 3.1. The Program Structure of RTCAS

The main program controls the sequence in which each one of the "Managers" will execute their tasks.

A single data structure contains all the information about the simulated tracks, the searchers, the environment and the simulation control required to run the program.

All units share and perform their tasks in a common data structure. The units could be changed provided their inputs and outputs remain the same, thus allowing future expansions or modifications in the software.

The implementation of the Data Fusion Model by the main program RTCAS is described in the algorithm shown in Figure 3.2. The code of RTCAS is in APPENDIX A1.

RTCAS general algorithm

```
Input initial simulation data
Input initial searchers data
Loop until selected time
      Get information from Searchers
      Loop for all searchers
            If searcher is in contact
                  Update for Positive Information
            else
                  Update for Negative Information
            end
            Normalize Tracks' Weights with Bayes Theorem
            Update Traks' movements
      Update cells probabilities with Total Probability Law
Output Probability Map
```

Figure 3.2. The algorithm implemented in the main program of RTCAS organizes the sequence or the component subprograms.

B. DATA STRUCTURE

1. Tracks Data

The number of tracks was fixed at 500 for computational convenience. Processing two searchers during 150 minutes of activity in the Local Area took 2 minutes running in a Personal Computer with a 486 DX2 66 MHZ processor, which implies a time compression ratio of 1/75. Any future increase in computational speed may permit an increase in the number of simulated targets in order to augment the statistical richness of the simulation.

Each track is represented by a record with fields for position, course and speed and weight. Information common to all targets is stored out of the records for memory space convenience, e.g., standard deviation in the initial positions, speed and course ranges, reaction distance and reaction turn angle.

All the information concerning the tracks is input by the user through the InterManager's corresponding procedures.

2. Searchers' Data

The program can process up to 15 searchers. However it is very unlikely that such a large number of assets will be concentrated for a Local Area or Lost Contact search.

Each searcher is represented by a record with fields for

position, course, speed, sonar parameters like frequency, status, source level, etc. and information regarding the contact.

The searchers' data structure is composed of static data like sonar frequency, detection threshold, etc., input by the user and controlled by the InterManager unit and by dynamic data like position, speed, sonar range selected, etc. read from a file by a procedure in the ScoutManager unit. Reading from a file resembles receiving the dynamic searchers' information through a data link or from the main stream of a tactical data system.

3. Environmental data

Input by the user through the User Interface includes bottom type, sound velocity profile type, wind speed and depth. This part of the data structure is used by DetecManager in the determination of the transmission losses.

4. Simulation control data

This part of the data structure stores information required in the control of the program like the name and location of input/output devices, (files in this implementation), times for which probability maps are desired, and delay in the start of the search from the datum time. This information is input by the user using Intermanager procedures.

C. ALGORITHM STRUCTURE

This section outlines the algorithmic structure executed by the units Interemanager, TargetManager, ScoutManager, DetecManager and CellsManager following the sequence established by the main program RTCAS to follow the Data Fusion Model.

1. Interface Manager

This unit declares the complete common data structure, creates the user input or static part of that structure and provides control of the program by means of a series of interactive menus called the User Interface.

For computational convenience, inside the program the calculations are done in measurement units different from those input by the user. The Interface Manager converts the data from the external to the internal unit system, i.e., discrete time unit (dtu), pixels, and discrete speed units (dsu), according to Table 3.1. The code for this unit, the file INTERMAN.PAS is in APPENDIX B and the sequence of menus is included in APPENDIX C.

	Internal Units	External Units
Time	1 dtu	5 minutes
Distance	1 pixel	0.1 Nautical Miles (185.2 meters)
Speed	1 dsu	1.2 Knots

Table 3.1. The measurement units conversion.

2. Target Manager

This unit implements the model described in Chapter II, Section A for the generation, updating, and reaction of the simulated targets. Figure 3.3 depicts the process to decide an evasion course at the target. Note that an evasion course is calculated for each searcher (*evasion*) and the one corresponding to the closest searcher (*d(s)<d(s-1)*) is selected as course. The code for this unit, the file TARGTMAN.PAS, is in APPENDIX D.

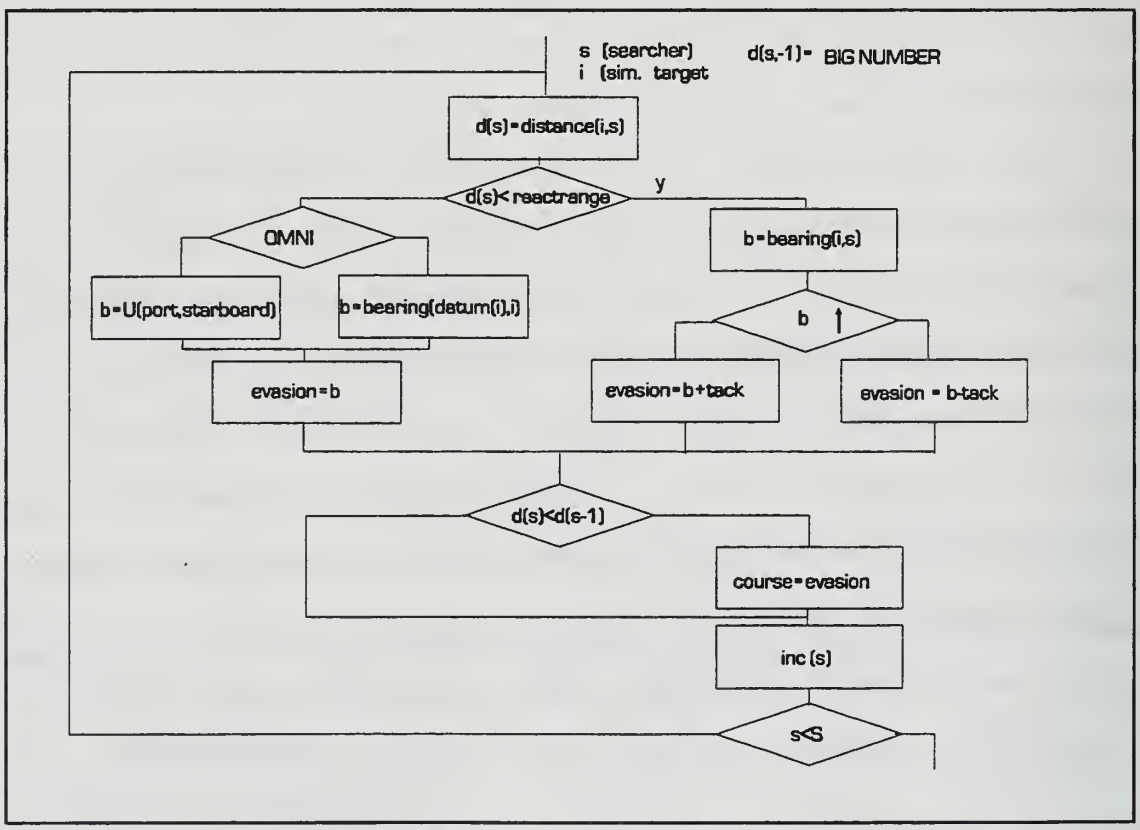

Figure 3.3. The determination of the evasion course at the target.

3. Scout Manager

Implements the core of the Data Fusion Paradigm as described in Chapter II section B.

The public procedures InitScouts, CloseScoutsDataLink, ConnectWithScouts and ReadScoutDataLink have the purpose of controlling the input of information from the searchers. In a real world implementation this procedure would receive the information through a Data Link device or from the main stream of the Tactical Data System of the platform. In RTCAS the information is read from a file with a format shown in Table 3.2.

ID	SEARCHER LOCATION		MOVEMENT		SONAR ON	SCALE	HOT	CONTACT LOCATION		MOVEMENT		CREDIBILITY	time
S1	214	205	336	10	1	10	0	-	-	-	-	-	25
H1	195	202	171	0	1	7	1	200	250	120	7	0.75	
H2	195	280	271	90	0	-	-	-	-	-	-	-	
S1	204	225	336	10	1	10	0	-	-	-	-	-	30
H1	195	202	171	0	1	5	1	205	245	120	7	0.85	
H2	195	280	271	0	1	5	1	207	252	150	4	0.75	
..	...	...	...	.	.	.	.	...	...	...	.	....	

Table 3.2. The Data Link information simulated by the content of a file.

The file, resembling the format of a data bus in a communications link, has a row for every searcher at every time updating period, fixed in the program to 5 minutes. Each row contains information regarding the identification of the searcher as a ship, helicopter or sonobuoy, position, course and speed, sonar state, sonar range scale selected and

contact information like position, course, speed and credibility. Course and speed are used to keep track of searchers as well as contacts and also used in the calculations of the transmission losses by DetecManager. For example, in Table 3.2, one surface ship and two helicopters are in the Local Area. At time 25 the ship S1 is emitting but does not have a contact, whirlwind H1 is dipping with its sonar, holds a contact and assigned a credibility value of .75. Helicopter H2 is in transit at 90 knots. The inclusion of the sonar conditions in the "data link message" makes the program generic for any type of platform. All that is needed to fuse the information with the simulation is position and whether or not the sonar is on or not. The identification at the beginning of the row is required to display an appropriate symbol. The value of credibility may be considered as analogous to the standard confidence level in the Allied doctrine (NATO, 1990).

The procedure ReadScoutsDataLink updates all fields in the searcher data base and when a contact is produced, the corresponding fields for contact information.

The basic structure of the Data Fusion Paradigm performed by the ScoutManager is shown in Figure 3.4.

The code for ScoutManager is in files SCOUTMAN.PAS and is shown in APPENDIX E.

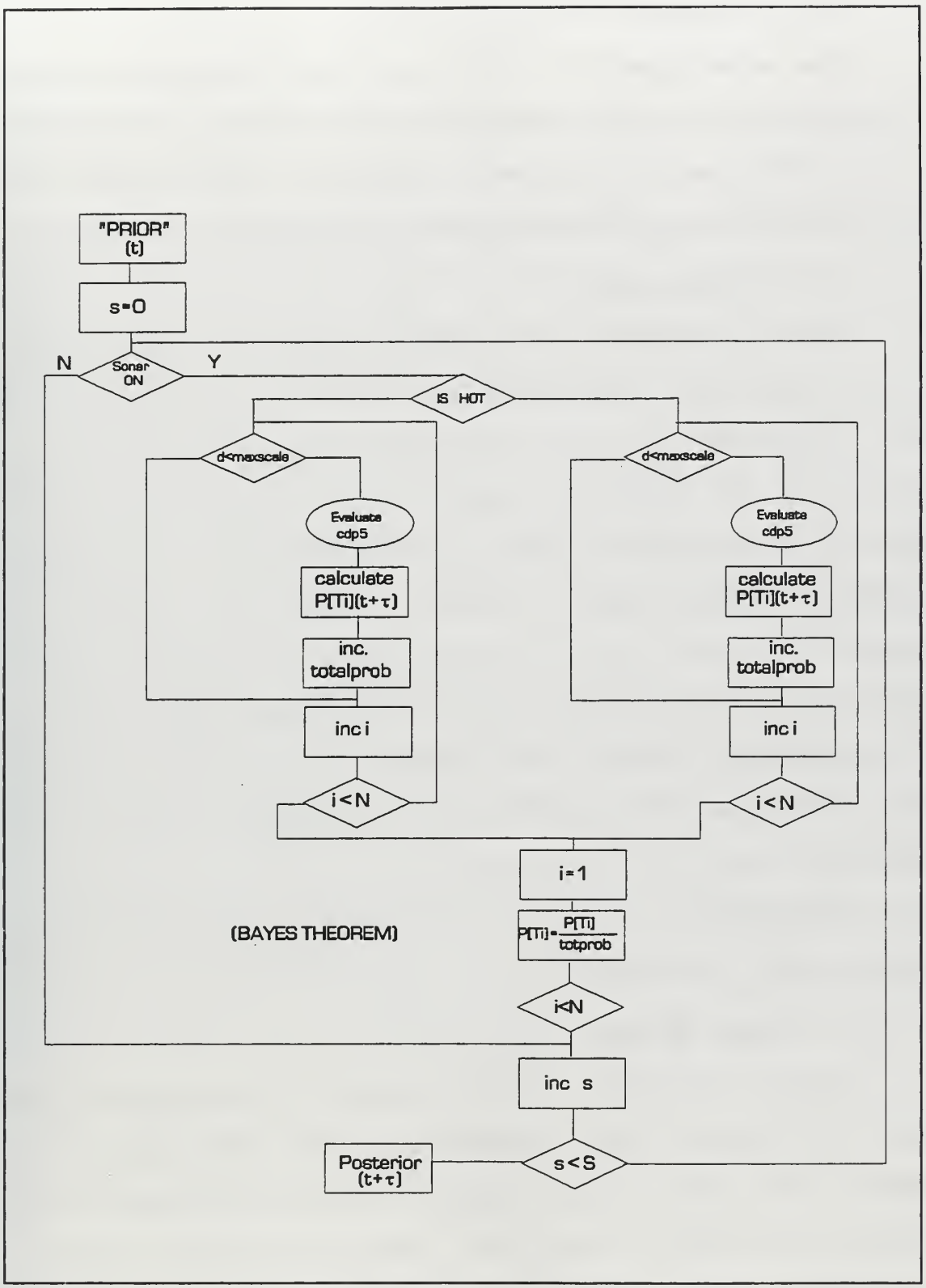

Figure 3.4. The basic algorithm for updating the prior.

4. Detection Manager

The sole purpose of this unit is the calculation of the value of cdp5, the five minutes cumulative detection probability as developed in Chapter 2, Section C. The code for this unit is in the file DETECMAN.PAS and is shown in APPENDIX F.

5. Cells Manager

This unit renders the Probability Map. The final product of the procedure UpdateProbinCell is a bidimensional array with a probability value associated with each cell. This product is the probability map. It could be used in a different kind of graphics interface or transmitted by the procedure BroadcastProbabilityMap. Figure 3.1 shows this output from CellsMan. In RTCAS the procedure outputs its broadcasting to a file that, in a real world application, would be replaced by an Input/Output device. The purpose of broadcasting is to provide the same information about the probability map to all searchers and other recipients.

The code for this unit is in the file CELLSMAN.PAS and can be seen in APPENDIX G.

6. Graph Manager

This unit produces the final output of the program in the form of a color coded Probability Map. Additionally, it produces the rest of the Graphic Information Display (GID) elements like grids, color scale, static and dynamic information display, etc. Figure 3.5 show the layout of the GID.

The colors in the probability map are calculated by the procedure SetProbabilityColors using the color scale shown in the lower left corner of the GID. The colors are relative to the largest probability value on any cell.

Instead of using one of the natural color scales like RGB or RYB (Elliot et al., 1995), white was selected as the brightest color in correspondence with the maximum probability value because of the black background RTCAS uses.

The GID shows the local area with the probability map together with graphic information regarding the searchers activity. The symbology found in most Tactical Data Systems was adopted to represent the different types and conditions of searchers and contacts.

The contact information is displayed with a hostile submarine symbol in the map and with information regarding the time of initial contact, course, speed, searcher holding it, etc. in the bottom of the screen.

The code for this unit is in the file GRAPHMAN.PAS and can be seen in APPENDIX H.

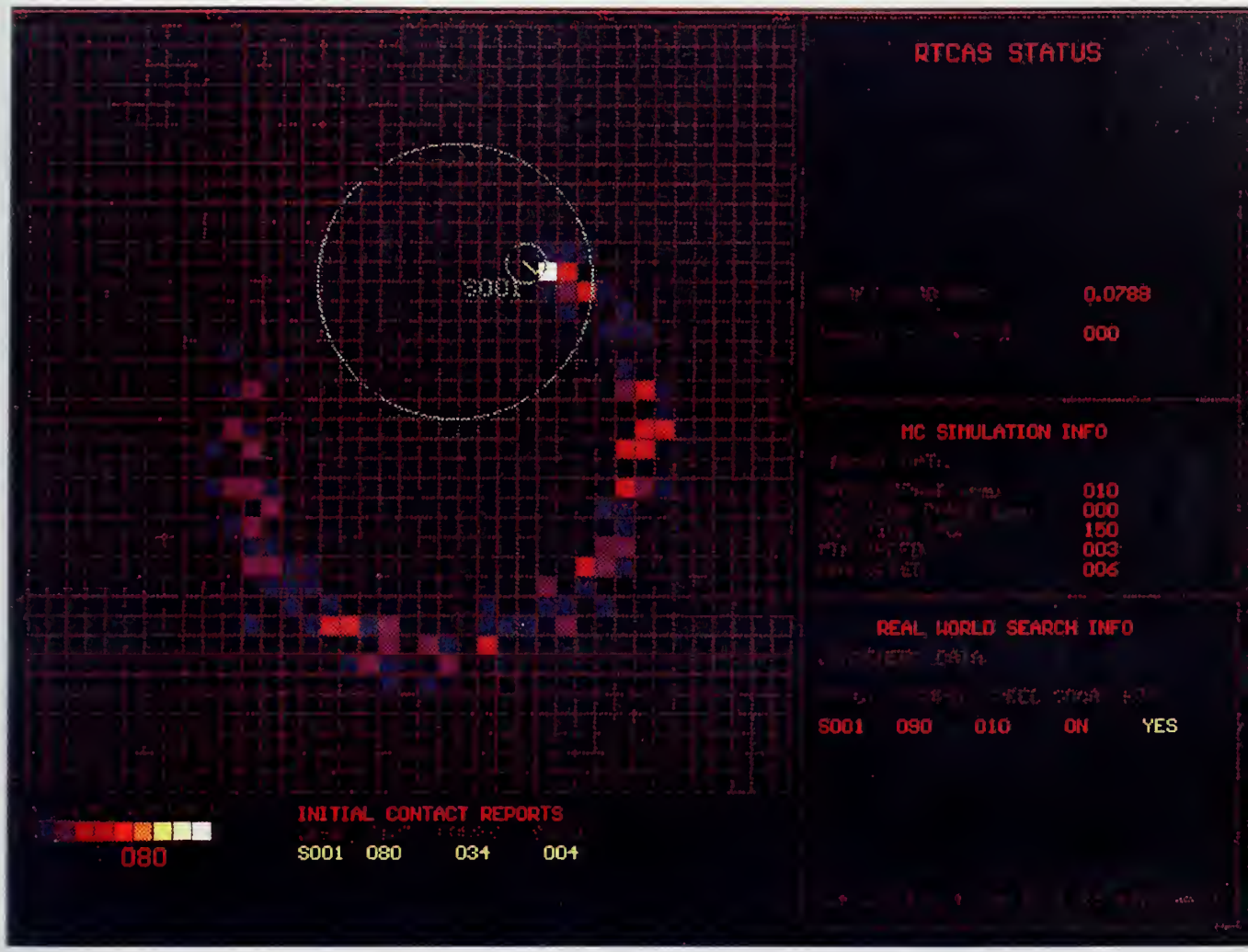

Figure 3.5. The Graphic Information Display (GID).

IV. DATA ANALYSIS METHODS: VERIFICATION AND EVALUATION OF THE MODEL

The Data Analysis Methods described in this chapter numerically verify the correctness in the implementation of the Bayesian Filtering Model and evaluate the significance of using a Reactive Target Model as compared to using a Non Reactive one.

The final product of the model is a probability map, hence the correctness of the model implementation as well as the relative efficiency of different methodologies must be evaluated on the model's ability to accurately construct the probability distribution of the location of the target.

The verification consists of:

- Selecting a "dummy," assumed to be the real target, among the simulated ones.
- Running several scenarios with different sets of variables.
- Extracting the test statistics of interest.
- Calculating the Measures of Effectiveness (MOE) based on those statistics
- Executing a test to verify a given hypothesis.

A. MEASURES OF EFFECTIVENESS

Accuracy, Area of Uncertainty, and Mean Missed Distance were selected as MOE's to respectively verify the correctness, dispersion and location of the distribution represented by the

probability map and the significance of a Reactive Target Model with respect to a Non Reactive one.

Mean Detection Probability and Mean Time to Detection were selected to evaluate the performance of the search conducted with a Reactive Target Model as compared to a Non Reactive one when the information provided by the probability map was used in conducting a search. The rest of this section describes each one of the selected MOE's.

1. Accuracy

Accuracy is a statistical measure of how well the probability map represents the model's uncertainty in the target's location (Widdis,1995,pp. 14). Furthermore, accuracy is a measure of correctness in the implementation of a Bayesian Updating algorithm. The containment statistic X is the cumulative probability of all cells G_{ij} with a higher likelihood than that of the cell G where the dummy is.

$$X = \sum_{(i,j)\in K} C_{ij} + U \sum_{(i,j)\in K'} C_{ij} \qquad \textbf{(4.1)}$$

where

$$K : \{\ (i,j):\ P(G_{ij}\backslash I) > P(G\backslash I)\ \},$$

$$K' : \{\ (i,j):\ P(G_{ij}\backslash I) = P(G\backslash I)\ \},$$

and U is a uniformly distributed random variate over the interval [0,1]. The second term in (4.1) represents a uniform proportion of the all cells where there is a tie,i.e., the cell probability is equal to that of the cell where the dummy

is. Assuming a discrete extension of the Probability Integral Transformation (Dudewicz et al., 1988, pp674) already used by Stone et al(1991,pp. 12) and Widdis (1995,pp. 15), the random variable X should be uniformly distributed over the interval (0,1).

The MOE described in this Section is a discrete version of a one sample Kolmogorof-Smirnoff Test Statistic (Dudewicz et al., 1988,pp. 671). The K-S statistic represents the maximum deviation of the empirical distribution of X from the Uniform,

$$D=supremum_{x\in[0,1]}|x-\hat{F}(x)| \qquad (4.2)$$

where the second term is the sample cumulative distribution function of X. Accuracy is defined as (Stone, 1991,p. 12),

$$A=1-D \qquad (4.3)$$

It varies between 0 and 1, the larger the value the more accurate.

Performing the K-S test not only provides a measure of correctness of the model by means of the accuracy but also permits a pictorial evaluation of the behavior of the model. If the percentiles of the sample CDF of x_k are plotted against the percentiles of a uniform distribution, the discrepancy between both becomes apparent for all values in the interval [0,1]. The nature of those discrepancies characterize a tracker as pessimistic or optimistic.

An optimistic tracker constructs a "concentrated"

probability map,i.e., it is optimistic about its own ability to determine the location of the target. As a consequence, the position of the dummy target is more often than not far from to the center of the empirical distribution, thus yielding much more frequently larger values of the containment statistic *X* (Figure 4.1).

A pessimistic tracker yields a more "dispersed" probability map, hence the frequency of lower values of the containment statistic is larger. For a given location of the target, the containment statistic *X* will be stochastically larger in an optimistic tracker than in a pessimistic one. In Figure 4.1, assume that both the pessimistic and the optimistic trackers represent the same situation. The value of the cumulative probability or "probability mass" inside any given equi-area contour in the optimistic tracker would be greater than that in the pessimistic one. As a consequence, if a contact were produced, for instance, at any point in such a contour,e.g., the point indicated with a chevron in both maps, then the containment statistic X would tend to be greater in the optimistic tracker than in the pessimistic one (0.9 and 0.2 respectively). Such a tendency translates into the shape of the relative cumulative frequency distribution of *X*. The example in Figure 4.1. shows that a given intermediate value of *X* , for instance 0.6, is much more frequent in a pessimistic tracker,i.e., 82% of the time, than in an optimistic one,i.e., 18% of the time.

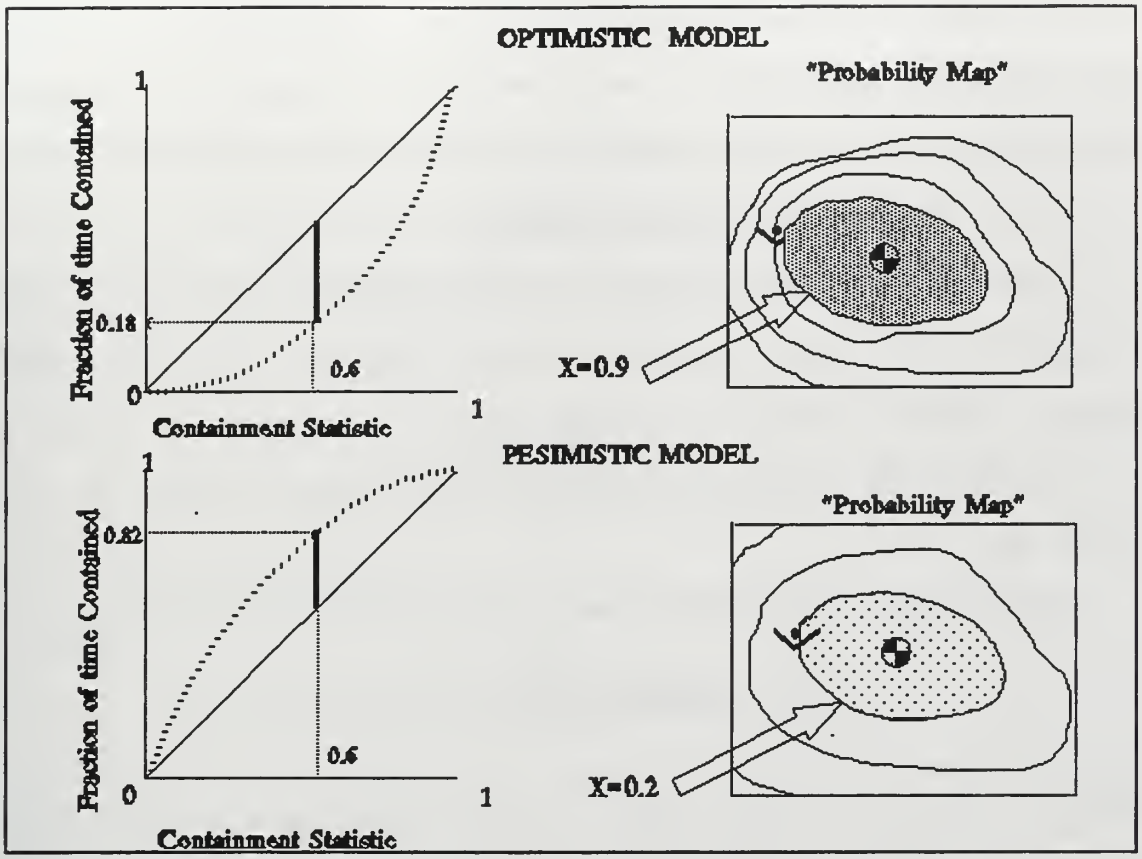

Figure 4.1. Examples of Pessimistic and Optimistic Trackers.

2. Area of Uncertainty (AOU)

The Area of Uncertainty (AOU) can be defined as the area of those cells (i,j) where $P(G_{ij}\backslash I) > P(G\backslash I)$,

$$AOU = \sum_{(i,j)\in K} a_{ij} \qquad (4.4)$$

where a_{ij} is the area of an individual cell and K is defined as in 4.1. The value of AOU provides a quantitative idea of how spread the probability map is. The smaller the AOU the better since the probability mass is more concentrated, thus leading to a better localization.

It may be argued that the AOU and the accuracy signify the same MOE, but it is clear that, for example, a constant value of accuracy may coexist with different values of AOU.

3. Mean Missed Distance(MMD)

The MOE selected to verify the location features of the probability map with respect to the dummy is the Root Mean Square Missed Distance, which can be expressed using a straightforward extension from the one used in Stone et al. (1988, pp. 14),

$$RMSMD=\sqrt{\sum_{i=1}^{N} p_i r_{di}^2} \tag{4.5}$$

where $p_i = P(T_i \backslash I)$ is the weight of each simulated target, N is the total number of simulated targets and r_{di} is the distance between the dummy d and each simulated target i. This MOE represents the standard error in the location of the target.

4. Mean Detection Probability (MDP)

The Mean Detection Probability is

$$DR=\frac{C_T}{N} \tag{4.6}$$

where C_T is the number of times out of N identical replications of an identical experiment in which a searcher obtains at least one contact within a period of time T.

5. Mean Time to Detection (MTD)

The Time to Detection TD is the period of time from the

beginning of a search, i.e., time zero of the simulation when the searcher enters the Local Area, to the time at which the first detection is produced or, if no contact is gained, the end of the experimental scenario. Then the Mean Time to Detection is the average among N replications of the experiment.

$$MTD = \frac{1}{N}\sum_{i=1}^{N} TD_i \qquad (4.7)$$

B. VERIFICATION OF THE MODEL IMPLEMENTATION CORRECTNESS

1. Test Structure and Experiment Design

The purpose of this experiment was to verify whether the implementation correctly models a Bayesian filtering process. The MOE or rather the measure of correctness was the Accuracy.

For this experiment a dummy target was selected at random among the 500 simulated targets. The Accuracy was measured with respect to the location of the dummy target. The detections were sampled based on the five minute cumulative detection probability (cdp_5), corresponding to each pair searcher-dummy.

The model was tested using Updating for Negative information only and for Negative and Positive information. In the former case the simulation ceased at the first contact. The Reactive Target model was tested in combination with each updating type.

Six different scenarios or searcher patterns were played as portrayed in Figure 4.2. In the first two the searcher is stationary resembling sonobuoys or dipping sonars, in the second pair the searcher moves through the datum area in a straight line, and in the fifth and sixth, one and two searchers respectively proceed with a random walk type of search around the datum.

The number of replications was 50 and for each one 20 time samples were taken. The K-S test requires a random sample,i.e., the observations should be independent and identically distributed (Dudewicz et al.,1988, pp.280). In this case, the 1000 time samples are not independent. Though only one sample could have been taken from each replication, all time samples were retained for they provided smoothing within a replication "batch".

The tests were identified with a 3 digit label. The first digit in the test label indicates whether Updating for Positive information took place, 1 if yes, 2 if not. The second digit denotes the Target Speed, 1 for slow (2-5 kts) and 2 for fast(6-12 kts) the last digit corresponds to the scenario.

The test structure is shown in Table 4.1 and the set of default values for the variables used in the experiment are shown in Table 4.2.

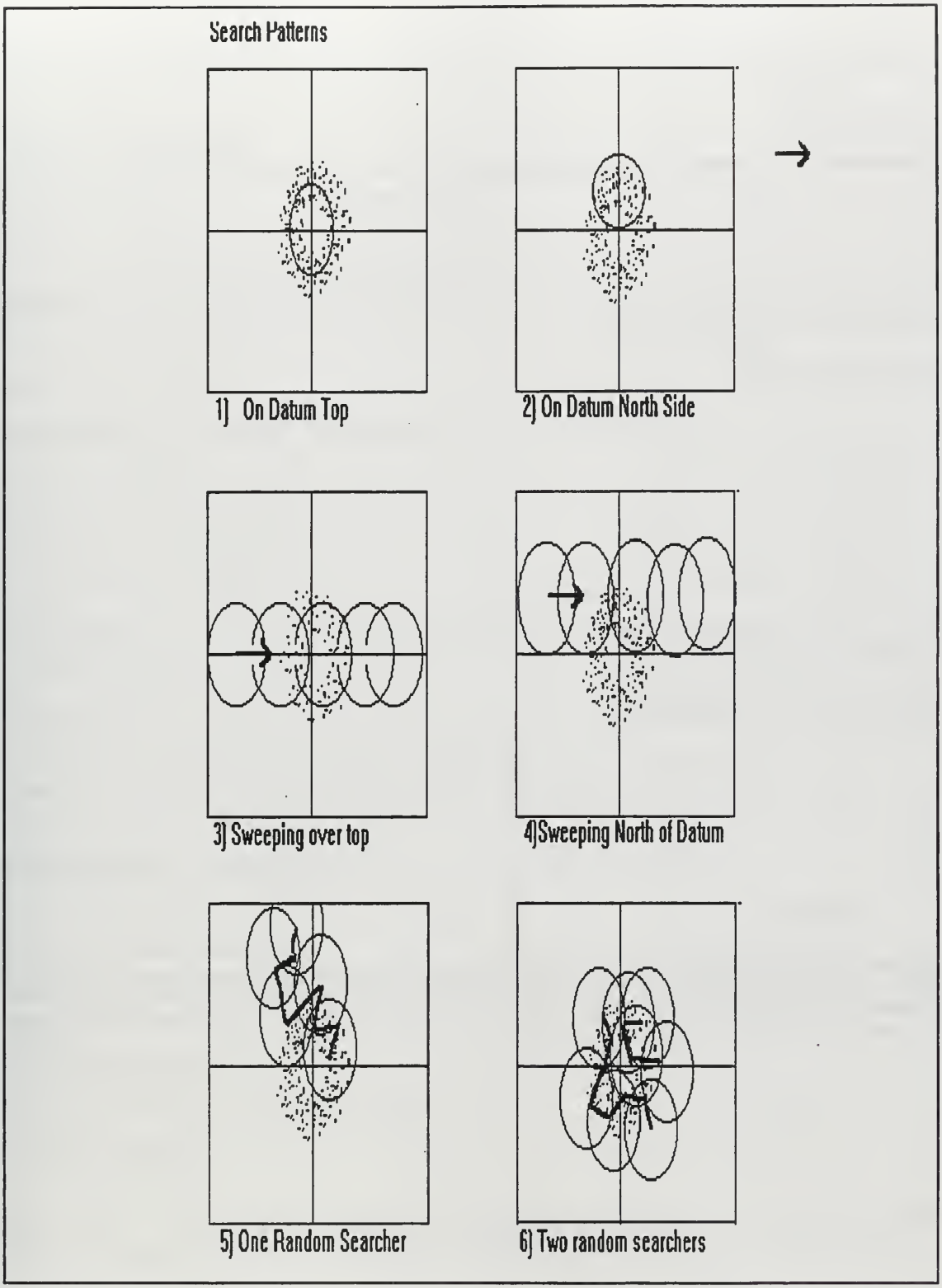

Figure 4.2. The search patterns utilized in the tests.

	SLOW TARGET (2-5 KTS)
NEGATIVE INFORMATION UPDATING ONLY	11
	12
	13
	14
	15
	16
POSITIVE AND NEGATIVE INFORMATION UPDATING	11
	12
	13
	14
	15
	16

Table 4.1. The Experiment Structure for testing the correctness in the implementation of the Bayesian Updating. In all Tests the Target Model and the Dummy are reactive.

VARIABLES	VALUE
TARGET VARIABLES	
Datum error	2 nm
Course Range	0-360°
Minimum Estimated Target Speed	3 kts
Maximum Estimated Target Speed	7 kts
Reaction Range	5 nm
Reaction Tack	150°
SEARCHER AND SONAR VARIABLES	
Source Level	200 Db re μPa
Ambient Noise Level	80 Db re μPa
Detection Threshold	20 Db re μPa
Directivity Index	30 Db re μPa
Frequency	5 Khz
Pulse Length	0.05 sec
Beam Width	12°
Circular Error in Contact Reports	1 nm
Overall Standard Error in Sonar Terms.	6 Db re μPa
Credibility	1
ENVIRONMENT VARIABLES	
Type of Bottom	Flat,Sandy
Sound Speed Profile	Isothermal
Depth	100 m
Wind	10 kts
TIME VARIABLES	
Total Simulated Search Time	100 min
Scouts Delay	15 min

Table 4.2. The default values of variables used in the Experiment to test the correctness of the model.

The correctness in the implementation of the model for each scenario was decided using a Kolmorgorv-Smirnov One Sample Test. The null hypothesis regarding the sample CDF of the containment test statistic X_k,

$$H_o: F(x) = x \tag{4.8}$$

tested for all x against,

$$H_1: F(x) \neq x \tag{4.9}$$

is rejected at level α if

$$D_n > d_{n,\alpha} \tag{4.10}$$

where D_n is calculated with (4.2) and $d_{n,\alpha}$ is the asymptotic critical point corresponding to the sample size n, obtained from Dudewicz et al. (1988,pp.670).

The accuracy for each test was calculated and plotted for a visualization of the pessimistic or optimistic nature of the resulting probability map and then logged in a table.

2. Results

Table 4.3 displays the results of this test. The first two columns describe the tests regarding the type of updating that was allowed, the target model in use and the type of scenario played.

The third column is the sample size n used in performing the tests. The variations are due to the fact that in the case

of updating for negative information only, the simulation stopped at the first contact. This happened in a random fashion in time hence the number of collected samples for each replication was random too.

The fourth column shows the average nature of Optimistic, Pessimistic or Variable in the probability map produced at every time step in accordance with Figure 4.1.

ID #	UPDATING NEG	UPDATING POS	SAMPLE SIZE *n*	OPT PES	KS TEST	ACCURACY
11	Y	N	173	VAR	arr	0.8919
12	Y	N	879	OPT	rrr	0.9343
13	Y	N	671	VAR	aaa	0.9625
14	Y	N	918	VAR	arr	0.9538
15	Y	N	189	VAR	arr	0.9052
16	Y	N	98	PES	aar	0.8767
21	Y	Y	1000	VAR	rrr	0.9343
22	Y	Y	1000	OPT	arr	0.9269
23	Y	Y	1000	VAR	rrr	0.9684
24	Y	Y	1000	VAR	aaa	0.9842
25	Y	Y	1000	VAR	rrr	0.9243
26	Y	Y	1000	VAR	aar	0.9123

Table 4.3. The implementation correctness test results.

The fifth column in Table 4.3 shows the result of the Kolmorgorof-Smirnoff test of hypothesis defined in the previous section. The three letters indicate the acceptance of rejection of the null hypothesis for three different levels : 0.01, 0.05 and 0.10. The null hypothesis is accepted most of

the time for the first level and rejected by a slim margin, 0.0152 in the average, most of the time for the other two.

The last column shows the final values of Accuracy obtained for each test. The results range from 0.8214 to 0.9842 with an average of 0.8945.

C. VERIFICATION OF THE IMPLEMENTATION ROBUSTNESS

1. Test Structure and Experiment Design

A sensitivity analysis was performed on Accuracy as a function of Target Speed, Searcher Speed, Datum Age, Source Level, Sample Size and Credibility. The purpose was to have an idea of the range within which the model remained correct. For this experiment, with the exception of the variable of interest, the conditions in test 25 (Table 4.2) were used for all tests.

2. Results

Figure 4.3. shows the plots of Accuracy as a function of Target Speed, Searcher Speed, Datum Age and Source Level. No conspicuous dependency can be observed within the explored ranges of each one of those variables, and accuracy remains high.

An increase in Accuracy as a function of Sample Size can be observed in Figure 4.4. This relationship conveys the idea that the quality of the tests is noticeably affected by the sample size.

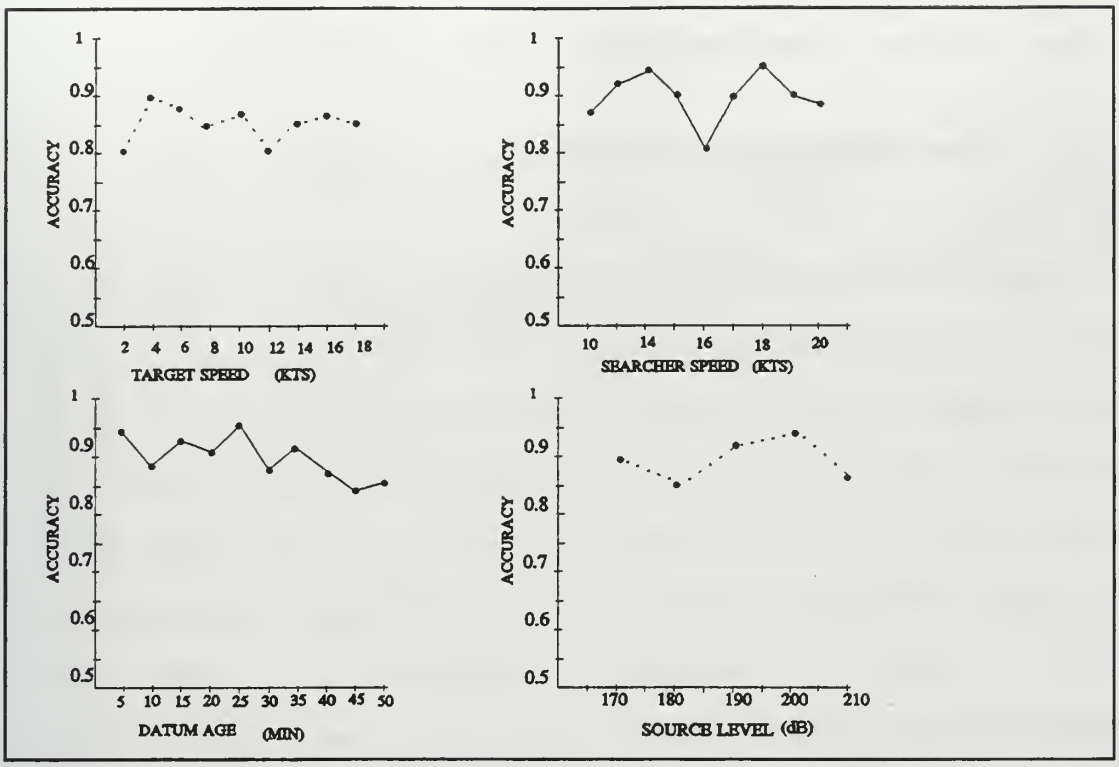

Figure 4.3. Accuracy as a function of Target Speed, Searcher Speed, Datum Age, and Source Level.

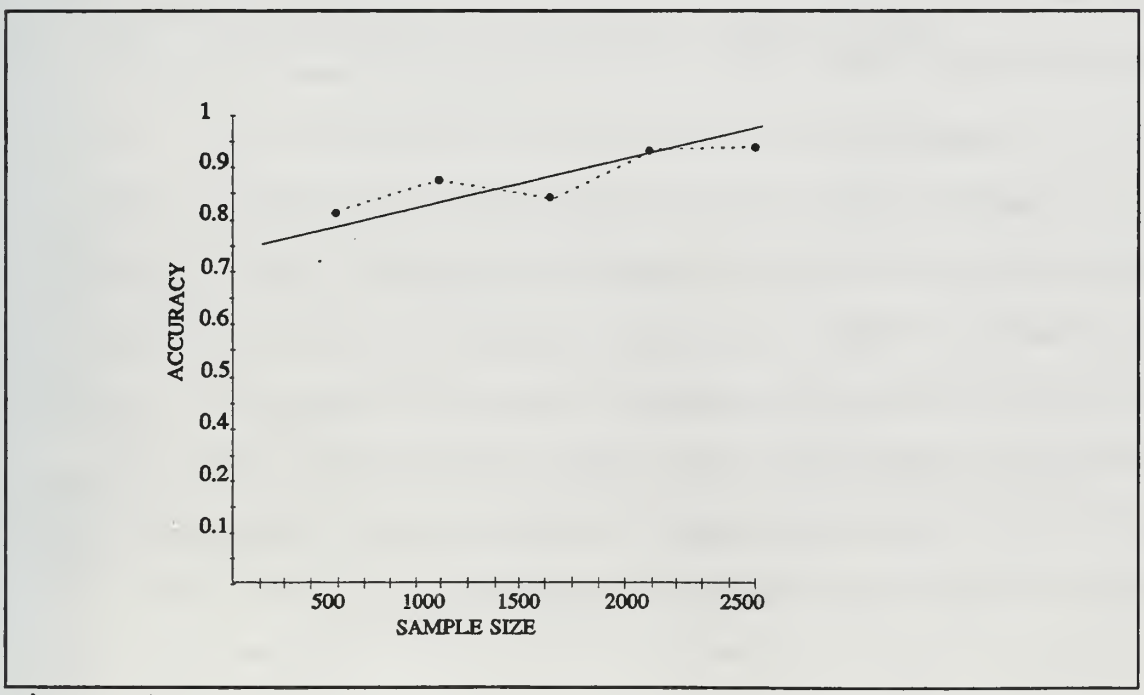

Figure 4.4. The quality of the test is sensible to the sample size.

D. THE INFLUENCE OF CREDIBILITY

1. Test Structure and Experiment Design

A sensitivity analysis was also performed on Accuracy as a function of Credibility since the latter is one of the hardest parameters to estimate. The purpose was to have an idea of the range within which the model remained correct. In this experiment the conditions in test 25 (Table 4.2) were used, except for the credibility *Q*. In the tests performed in Section B, credibility was set to be equal to 1. In RTCAS, the user is allowed to input other values. This experiment tested values of Q between 0.6 and 1. Since in this test, as in Sections B and C, the contacts were generated only at the position of the dummy, i.e., there were not false alarms, the true value of credibility in the simulation was always 1.

2. Results

As can be observed in Figure 4.5, the Accuracy curves were plotted for several values of Credibility larger than 0.5 since smaller values are very unlikely to be used. When Q decreases the Accuracy decreases as well. A tendency towards a pesimistic probability map output can be seen as decreasing values of Q causes the model to ignore good information and produce a less "concentrated" probability map.

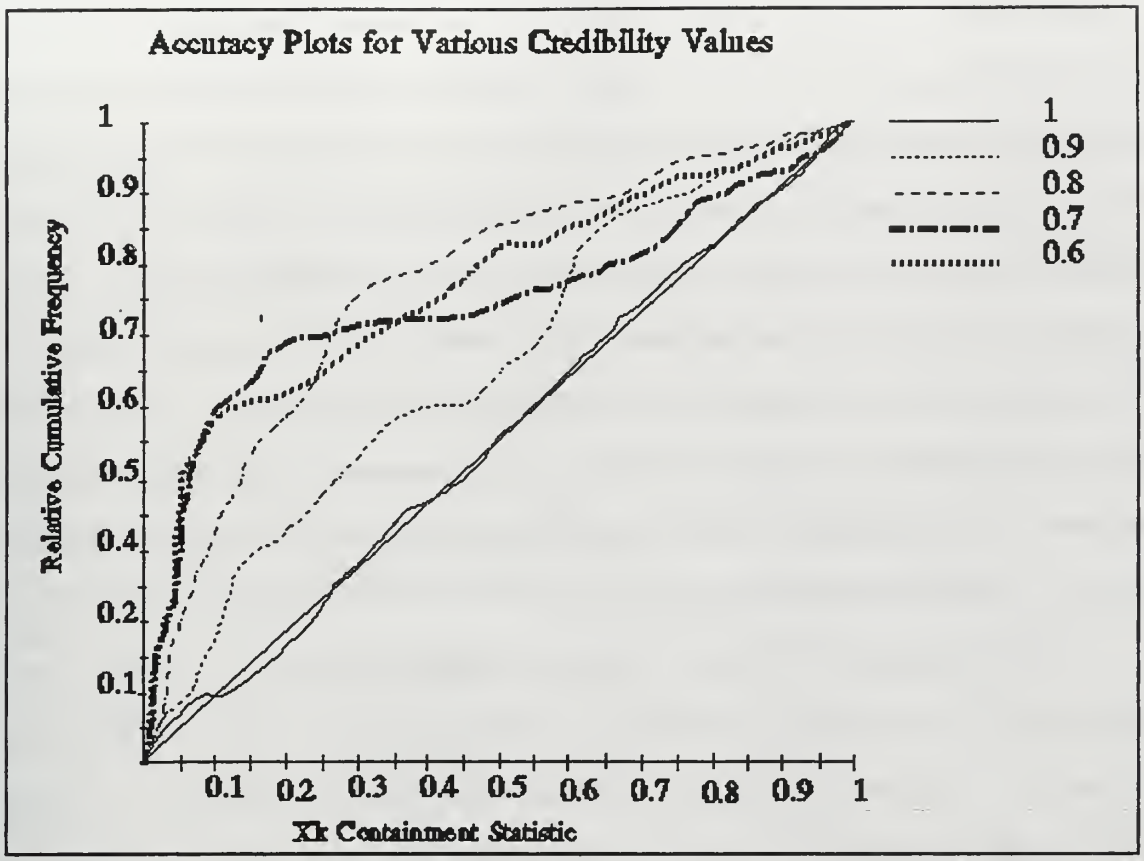

Figure 4.5. Decreasing the value of Credibility makes the probability map more pessimistic.

E. THE SIGNIFICANCE OF A REACTIVE TARGET MODEL

1. Test Structure and Experiment Design

The purpose of this experiment was to test whether or not a Reactive Target Model is operationally better than a Non Reactive one. All of the MOE's defined in section A were used in order to compare the results from both Target Models.

The dummy was selected among the five hundred simulated targets by means of a uniform variate drawn and was set to be always reactive. This implies the assumption that a real target is reactive. All MOE's were measured with respect to

the location of the dummy target. The detections were decided as in Section B. The model always updated for negative information only, stopping at the first contact. The purpose of not updating for positive information is that a contact report will noticeably improve most of the MOE's due to the "concentration" of the probability mass in the surroundings of the target thus obscuring the difference between a Reactive and a Non Reactive Target Model. Furthermore, the differences between both Target models are more pivotal to the problem before the first detection.

The Reactive Target Model was tested in the same conditions as the Non Reactive Target Model. Two different ranges of target speed and three different values of datum delay were tested as can be observed in the test structure shown in Table 4.4. The main purpose of the test was to compare how good each model (reactive or non reactive) deals with the same scenario when constructing the probability map. As a consequence variables with the greatest impact in the kinematics of the problem were selected.

The number of replications was the same as in the experiment in Section D. The tests were labeled with a 3 digit code. The first digit in this case indicates the speed range, the second the delay and the last one the scenario or search pattern, which are the same as in the test in Sections D and E with the addition of a seventh pattern representing the first 270 degrees in an exhaustive spiral search. The set of

default values for the variables used in the experiment remained the same as in Table 4.2 with the exception of the target speed and delay time which were varied from test to test. For each running all MOE's were recorded in separate files for the reactive and non reactive case and then compared by means of a Wilcoxon Test for paired data (Stone et al., 1991,pp.28). The Wilcoxon Rank-Sum Test (Devore,1991,pp.612) is a non parametric test used to compare paired observations without regard of the distributions they come from.

Target Speed/ Delay	2-5 kts	5-11 kts
0 min	111 112 113 114 115 116	211 212 213 214 215 216
15 min	121 122 123 124 125 126 127	221 222 223 224 225 226 227
30 min	131 132 133 134 135 136 137	231 232 233 234 235 236 237

Table 4.4. Test structure for comparing the Reactive Target Model with the Non Reactive one. Each one of the tests in the table was performed for each model.

Let R_i and N_i be the corresponding MOE's for the case of a Reactive and a Non Reactive Target Model respectively for each run $i=1..n$. We want to compare whether or not one of them has a tendency to be larger than the other. R_i and N_i are correlated for each i, but the differences $D_i = R_i - N_i$ are independent for all i; the Wilcoxon test makes their original distributions irrelevant to the comparison. A statistically valid statement about such a comparison can be done by means of a test of hypothesis of the mean μ_N and μ_R of each r.v. where

$$H_o: \{\mu_R - \mu_N = 0\} \tag{4.11}$$

and,

$$H_1: \{\mu_R - \mu_N > 0\} \tag{4.12}$$

are the null and alternate hypotheses. The order in the differences between both mean values can be varied depending on the nature of the MOE. For instance if R_i and N_i represented the Accuracy, we would want to know which is larger since the larger the better in such a case. Conversely if they were the Area of Uncertainty, we would want to find out which is smaller.

In this test, the differences $D_i = (R_i - N_i)$ are obtained and sorted by ascending absolute value. The rank of such a sorting is r_i. Adding up the rank of all negative D_i yields the test statistic

$$W = \sum_{D_i > 0} r_i \qquad (4.13)$$

If there is not a tendency of the random variables R to be smaller than N, then W is expected to be roughly close to the value of its mean $\mu_W = n(n+1)/4$, i.e., half of the sum of the ranks from 1 to n. Conversely, if R_i tends to be smaller, the test statistic W will be smaller than $n(n+1)/4$.

For $n>30$ W is a normal random variable with mean μ_W and variance $\sigma_W^2 = (n+1)(2n+1)/24$. As a consequence the test of hypothesis can compare the value of W with the critical value of a normal distribution $C(\alpha,n)$ corresponding to the sample size n and the desired significance level α. If $W > C(\alpha,n)$ then we reject the null hypothesis.

2. Results

Table 4.5. sumarizes the results of the Wilcoxon Tests of Hypothesis conducted to verify whether or not using a Reactive Target Model improves the performance of the tracker from the standpoint of Accuracy, Area of Uncertainty and Mean Missed Distance. The tests were performed on the data shown in Tables 4.6 and 4.7 , where for each MOE and each run, an asterisk indicates when the Non Reactive Target Model wins for a visual appraisal of the tendencies.

On the average, as can be observed in Table 4.5, second column, Accuracy, Area of Uncertainty and Mean Missed Distance show a better result for a Reactive Target Model. However,the

Wilcoxon Test does not reject the null hypothesis that there is no difference between the RT and the NRT models. Furthermore, the null hypothesis should be accepted for Accuracy and Area of Uncertainty. This result is somewhat disappointing for we expected the former to be the MOE where the improvement would be the greatest. Nonetheless, at a 10% significance, H_o would be rejected.

The RT model proved to be definitely better than the NRT one with respect to the Mean Missed Distance, yet by a slim margin.

	MEAN NRT	MEAN RT	W	μ_W	σ_W	p-value	H_o	TEST RESULT α= 1%
ACC	0.8292	0.8567	285	451	152	0.2743	$\mu_N-\mu_R=0$	Accepted
AOU	9.8121	8.3223	187	451	152	0.0834	$\mu_R-\mu_N=0$	Accepted
MMD	7.0542	5.8874	50	451	152	0.0084	$\mu_R-\mu_N=0$	Rejected

Table 4.5. Significance of a Reactive Target Model in the average and in a Wilcoxon Test of Signed Ranks. W is the Wilcoxon Test Statistic.

F. FURTHER EVALUATION OF THE SIGNIFICANCE OF A REACTIVE TARGET MODEL: A NON OPTIMAL MYOPIC SEARCH EXPERIMENT

1. Test Structure and Experiment Design

Although the experiment described in Section E ascertains that a Reactive Target Model performed better than a Non Reactive one in most of the cases, the difference in effectivness reflected by the tests is much less significant than the expectations. In the previous set of tests the searchers proceeded without regard to the probability map

output by the model. They followed typical lost contact exhaustive initial search patterns.

In order to further evaluate the significance of a Reactive Target Model as compared to a Non Reactive Target, an experiment was conducted in which the searchers were set to be fed back by the output of the model, conditioning their behavior by the probability map rendered at every time step. At all times, the single searcher simply headed for the brightest cell.

The visualization of the search pattern conducted with the above mentioned rules was often irrational from a tactical point of view in the sense that the searcher would, for instance, move from one brightest cell to the other leaving unexplored cells that were close to his way. Again, the purpose was to have a standard set of rules of behavior of the searcher in order to obtain more insight in the comparison between a Non Reactive Target Model and a Reactive one. The dummy, uniformly drawn from the total number of simulated targets at each replication, was set to be always reactive and the actions were set to last until a contact was obtained or 150 minutes, whichever occurred first. The detections were decided as in section B. The set of tests was again identified with a 3 digit code as shown in Table 4.8. The first digit indicates the target speed range, the second one the datum age and the third one the searcher speed and initial position.

ID #	ACC		AOU		MMD	
	NRT	RT	NRT	RT	NRT	RT
111	0.7523	0.8500	6.6238	4.4575	4.2770	3.4273
112	0.9361	0.9636	10.0275	5.9462	3.5910	2.5723
113	0.9295	0.9353	9.3438	6.5120	4.4159	3.9566
114	0.9518	0.9551	12.2125	9.5123	4.6987	3.6002
115	0.7795	0.8541	5.4262	2.5634	3.4049	2.7164
116	0.7401	0.8456	2.8725	2.1560	2.8419	1.2563
117	0.9089*	0.8687	10.3038	3.4521	3.8001	2.5800
124	0.7134	0.9635	7.9650	4.7425	4.9179	4.1629
122	0.9441*	0.8242	13.2612	10.2040	4.6058	4.2763
123	0.9230	0.9261	10.6025	9.5623	6.1347	5.4798
124	0.8035	0.9012	17.2188	12.9650	6.0323	5.1757
125	0.8872	0.9162	10.1212	8.5612	4.5867	5.3936
126	0.7731*	0.6647	3.8525	2.3350	3.6402	2.8612
127	0.8810	0.8623	11.618*	12.2564	5.3115	4.7462
131	0.6532	0.7823	5.1925*	5.5478	5.5896	4.5200
132	0.8452*	0.8124	9.5412	8.4150	5.9528	3.5600
133	0.8687*	0.8536	11.2462	9.9871	6.6959	6.4589
134	0.8049	0.9304	17.3588	15.2412	7.2086	6.0514
135	0.7689	0.8120	7.4200	5.8650	5.6523	4.0254
136	0.8897	0.7025	4.6812	2.7475	4.6841	3.8912
137	0.8873	0.9151	9.430*	10.2547	5.9379	4.2356
avg	0.8400	0.8637	9.3485	7.2992	4.95149	4.0451

Table 4.6. The results of the Comparison between a Reactive Target Model (RT) with a Non Reactive one (NRT) for slow target speeds. Asterisks indicate when NRT wins.

ID #	ACC		AOU		MMD	
	NRT	RT	NRT	RT	NRT	RT
214	0.6947	0.7942	7.7325	5.7474	6.5314	4.1247
212	0.896*	0.7123	13.0422	5.4373	6.5159	2.8195
213	0.8624	0.9237	9.6524	8.4512	6.9158	6.1266
214	0.893*	0.8222	12.2575	8.5647	7.9634	4.9579
215	0.7242	0.8520	7.6672	7.6542	5.2720	3.2549
216	0.733*	0.7047	3.3073	2.0707	4.0580	3.1092
217	0.8962	0.9014	13.1071	8.4475	5.9064	3.6117
221	0.7530	0.8562	12.5350	9.4358	10.0125	8.6542
222	0.8044	0.9115	8.445*	8.4512	8.4972	5.7633
223	0.9134	0.9139	13.2425	9.5623	9.8595	9.1776
224	0.901*	0.8425	13.4175	12.6275	9.7686	8.3196
225	0.8223	0.8366	11.8777	7.5158	8.8233	7.1634
224	0.7219	0.8675	4.050*	5.6475	7.7192	6.1558
227	0.926*	0.9027	12.025*	12.9775	9.8413	6.2208
234	0.8125	0.8457	19.6851	10.2125	13.4794	11.8224
232	0.7846	0.9315	6.267*	8.6958	10.9819	10.2591
233	0.903*	0.8728	8.965*	9.8561	12.9581	12.8511
234	0.8044	0.8841	9.795*	15.6056	12.9458	12.7729
235	0.7337	0.8412	6.325*	12.7345	9.817*	11.4833
236	0.7951	0.8324	15.1725	7.4125	12.1102	12.0041
237	0.7711	0.7951	6.962*	20.1525	12.3220	11.6747
AVG	0.81849	0.84975	10.26347	9.345557	9.157086	7.729829

Table 4.7. The results of the Comparison between a Reactive Target Model (RT) with a Non Reactive one (NRT) for fast target speeds. Asterisks indicate when NRT wins.

TEST ID	TARGET SPEED	DATUM AGE	SEARCHER SPEED
111	2-4 Kts	10 min	12 kts
112	2-4 Kts	10 min	18 kts
121	2-4 Kts	60 min	12 kts
122	2-4 Kts	60 min	18 kts
211	4-8 Kts	10 min	12 kts
212	4-8 Kts	60 min	18 kts
221	4-8 Kts	10 min	12 kts
222	4-8 Kts	60 min	18 kts
311	8-12 Kts	10 min	12 kts
312	8-12 Kts	60 min	18 kts
321	8-12 Kts	10 min	12 kts
322	8-12 Kts	60 min	18 kts

Table 4.8. The non optimal myopic search experiment structure.

For each test 30 replications were run and the average values of Detection Rate, Time to Detection and Search Speed were extracted.

2. Results

Table 4.9 shows the results of this test. The Mean Detection Probability showed the strongest difference between a Non Reactive model and a Reactive one, favorable to the latter. The Wilcoxon Test of Signed Ranks Sum performed on Mean Detection Probability and Mean Detection Time yielded the results shown in Table 4.10. At a 5 % significance level, the null hypothesis that there is no difference between a Reactive Target Model and a Non Reactive one can be Rejected with

respect to the Mean Detection Probabilities and should be accepted regarding the Mean detection Time.

The observation of the evolution of the search step by step on the screen output of the model, provides more insight about the real gist of Mean Time to Detection in the context of the test. In the Non Reactive Target Model, the searcher promptly starts "consuming" probability mass since the expanding ring of the probability map does not change because

	MDP		MDT	
	NRT	RT	NRT	RT
111	0.700	0.900*	73.8	54.3*
112	0.767	0.933*	28.1*	28.3
121	0.267	0.700*	48.8	39.6*
122	0.267	1.000*	55.6	32.1*
211	0.267	0.167*	75.5	67.5*
212	0.667	0.667*	45.1	39.2*
221	0.167	0.267*	45.2*	46.1
222	0.833	0.900*	61.3*	63.3
311	0.000	0.000	150.0	150.0
312	0.567	0.633*	54.2*	56.7
321	0.000	0.000	150.0	150.0
322	0.267	0.367*	58.6	39.6*
AVG	0.431	0.486*	45.5	38.9*

Table 4.9. The non optimal myopic search experiment structure .

MOE	W	$\bullet_W$	o_W	p-value	H_o	H_1	RESULT
MDP	0	39	15.7	0	$\mu_R=\mu_N$	$\mu_R>\mu_N$	Rejected (5%)
MDT	18	39	15.7	0.059	$\mu_R=\mu_N$	$\mu_R>\mu_N$	Accepted (5%)

Table 4.10. The Reactive Target Model performed better than the Non Reactive one with respect to Mean Detection Time but no difference was appreciated with respect to Mean Detection Probabilities.

of the searching activity. Therefore, when it proceeds towards the brightest cell, it tends to have a lot of simulated targets within its detection range. This is reasonable since the relative velocity between each simulated target and a searcher will be high because of the average outbound nature in the course of the latter and the inbound course in the former (See Figure 4.6). Conversely, in the Reactive Target Model, the presence of the searcher will "deform" the expanding ring of the probability map in a way such that the relative velocity between searcher and targets will be low in general, i.e., a chasing situation will be the norm most of the time. As a consequence, the rate of consumption of probability mass will be lower than in the Non Reactive case and the time to detection will be larger, thus leading to a slower Search Speed. (See Figure 4.7)

Coalescing the ideas of the three MOE's analyzed in this section, it can be stated that using a Reactive Target Model yielded a slower yet more effective search. Nonetheless the margins by which this sentence can be stated are rather slim.

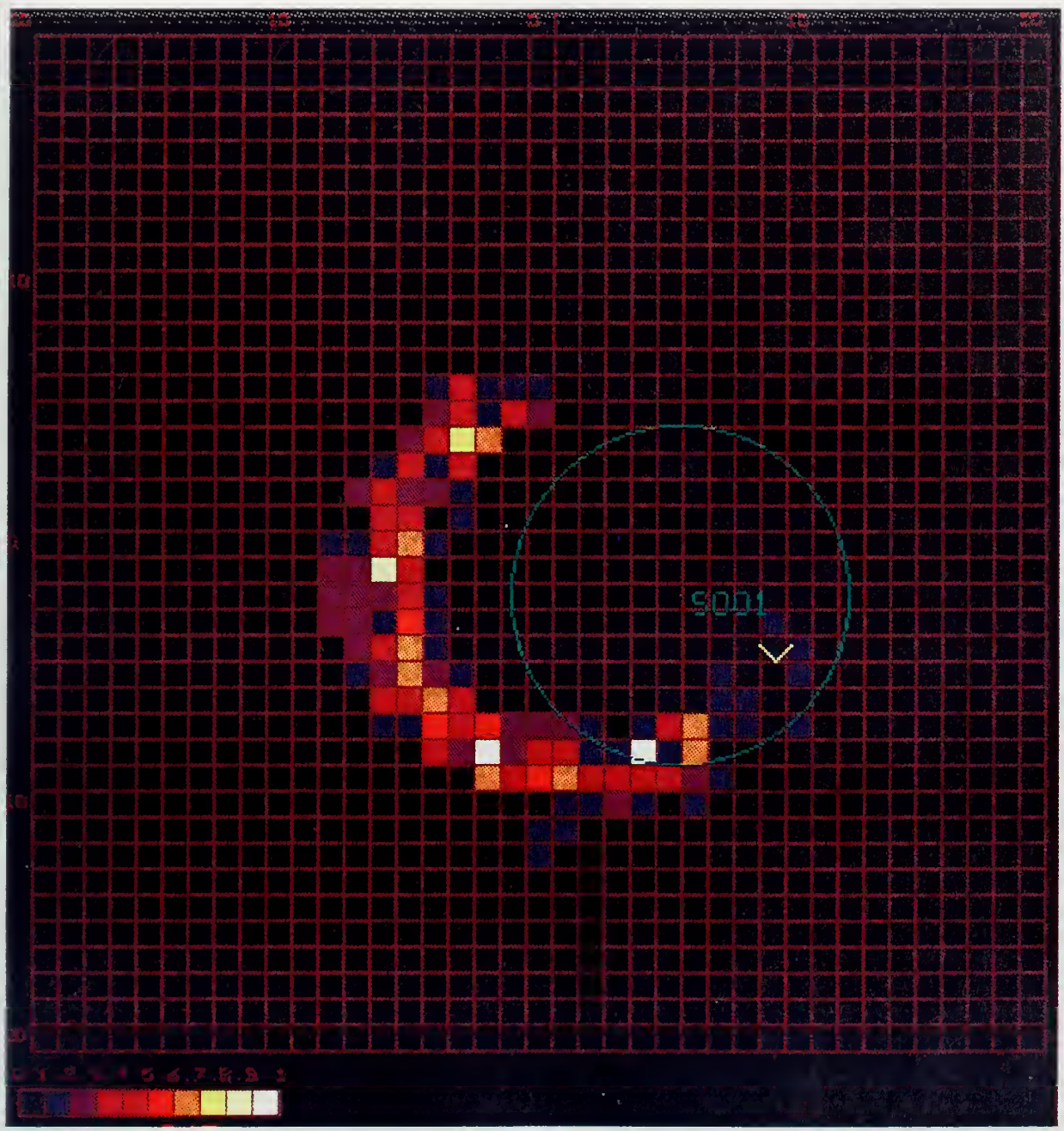

Figure 4.6. The searcher "consumes" probability mass to the N.E. of the expanding datum of a Non Reactive Target Model. The darker colors indicate those cells where the likelihood is reduced by the search.

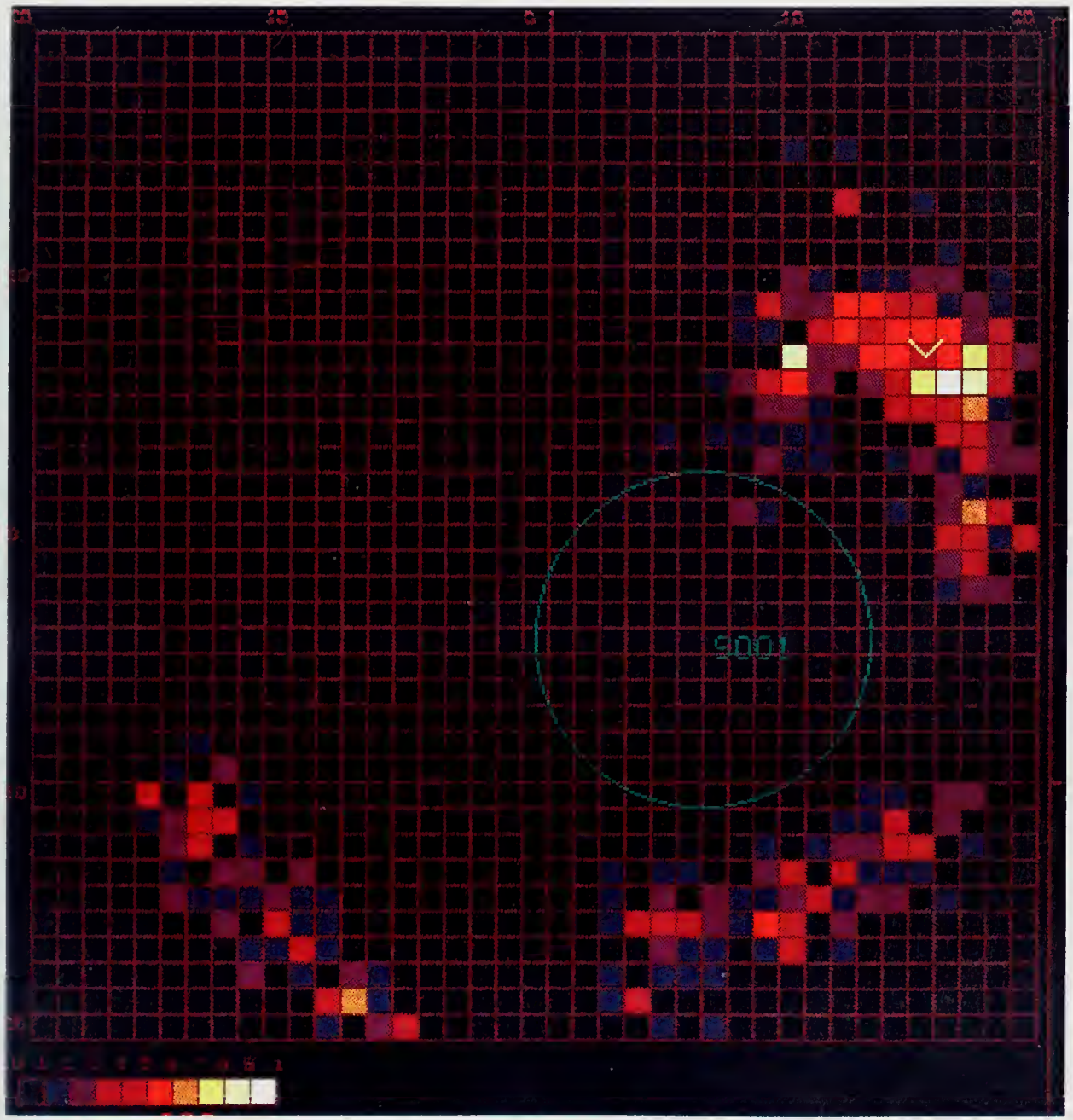

Figure 4.7. The reaction of the target prevents the searcher from "consuming" too much probability mass as it proceeds with the search pattern. The expansion of the datum is somewhat irregular.

V. EXTENSIONS OF THE CONCEPT AND IMPLEMENTATION ENHANCEMENTS SUGGESTED FOR FUTURE RESEARCH

A. EXTENSIONS OF THE CONCEPT

The Information Fusion Paradigm (IFP) can be the engine of several problems other than the one solved in this thesis.

1. Other Sensors

Extension of the model to other sensors like radar, lidar, infrared, etc. is straightforward. Even the idea of a multisensor search is plausible provided that the corresponding models to evaluate the effectiveness of the search are implemented. The structural design of the program allows the interchange of the sonar detection model almost without penalties.

2. RTCAS and Java

The appearance of the programming language Java (Sun Microsystems Corporation, 1994) offers a new dimension to the Information Fusion Paradigm. Java allows relatively easy networking between different types of computers. This new level of interchange of information not only favors the handling of the real part in the IFP, but also brings the possibility of a Distributed Simulation (Buss, et al.,1996), as an option to deal with the "virtual" part of the IFP. Such a connectivity is a perfect solution when for instance the search is conducted with heterogeneous assets, as in the realm of the Search and Rescue (SAR) problem. According to interna-

tional regulations, any vehicle, e.g., fishing or merchant vessels, airplanes, lorries, etc. can be convoked to participate in a search by a SAR agency. Nowadays, the only existing links between those heterogeneous assets are the International Maritime Radio Circuits. Given that most of those platforms can carry a computer, Java brings the possibility of linking them together. Then, it is possible to implement a Real Time Bayesian Filtering in which each one of the "convoked" searchers will not only input its searching activity via, for instance, an Internet browser, but also will run its own Stochastically Synchronized Simulation, i.e., the seed of the random generation is broadcasted in the network. Hence each one of the searchers will have its own picture of the situation thus leading to a more intelligent allocation of effort.

B. IMPLEMENTATION ENHANCEMENTS

1. Model enhancements

We envision a lot of potential growth without the penalty of very expensive research and development. Some ideas susceptible to further research are outlined in this section.

A Multiprior target generation scheme would enhance the ability of RTCAS to capture reality by means of several weighted options.

An Intelligent Reactive Target Model would allow each target to evaluate more factors before choosing a Reaction

Pattern. One of those factors may be the counterdetection estimated range. Solving the sonar equation for the one way path from the searcher to the target is more realistic than using a fixed distance as a reaction parameter. Artificial Intelligence tools and concepts would be in place when developing the new reactive features.

In water depths in the transition between deep and shallow from the acoustic point of view, shadow zones can be expected within several cases of sound velocity profiles. As a consequence, the search effort is not uniform over the water column. A possible solution to this problem would be enhance RTCAS with the addition of a third dimension in the data structure. The probability map could be rendered for different layers that would help decide how to allocate the search effort not only in time and xy positions but in depth as well.

2. Implementation Enhancements

Pascal was selected as a programming language because of its robustness and maintenance simplicity. However, RTCAS may benefit from implementation in other languages like C or C++ mainly because of the speed provided by the pointer arithmetic so useful in the simulation, or Java, because of the connectivity that comes with the language. The decision should be done only after evaluating advantages and disadvantages of such a change or expansion.

Regardless of the language, RTCAS must migrate to a graphic interactive user interface in order to allow easier manipulation of its possibilities. The more the interface of the program resembles other popular programs, the faster the learning curve and, as a consequence, the efficiency in the use of its resources. In the realm of the user interface, the need can be also mentioned for zooming and scaling on the fly.

VI. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

RTCAS correctly implements a Bayesian Filtering Process for Fusing Information from a real world active search with a simulation encompassing many possible positions and movements of a reactive target to produce a probability map of target location.

In an active search, the target will invariably react. Using a very rudimentary Reactive Target Model proved to be better than a non reactive one in most of the cases.

B. RECOMMENDATIONS

Intensive testing of RTCAS should continue to further verify its behavior and to validate it for use in real environments.

Extensions of the Information Fusion Paradigm should be explored and enhancements to RTCAS should be implemented to increase its power.

The possibility of implementing a reactive target model in currently used TDA's should be explored.

APPENDIX A. RTCAS MAIN PROGRAM

```
program rtcas;

uses
GRAPH, CRT,GraphManager,TargtManager,DetecManager,
CellsManager,Scoutmanager,InterManager;

begin
  clrscr;
  Graphmanager.Start;
  Scoutmanager.Initscouts;
  while WannaContinue do begin
    WannaSeeData:=FALSE;
    Scoutmanager.ConnectWithScoutsThruDataLink;
    Intermanager.AskOptions;
    Detecmanager.UpdateSonarParameters;
    if WannaContinue then begin
      Targtmanager.GenTargets;
      Graphmanager.Pleasewait;
      ScoutManager.ResetUndetectedMass;
      if WannaSeeData then begin
        Graphmanager.DisplayData;
      end else begin
        for time:=1 to totalsimtime do begin
          Scoutmanager.ReadScoutsDataLink;
          Scoutmanager.UpdatePriorWithScoutsInfo;
          Targtmanager.UpdateWeightsWithBayes;
          Scoutmanager.UpdateUndetectedMass;
          Cellsmanager.UpdateInCellProbabilities;
          Graphmanager.OutputGraph;
        end;
      end;
    end;
    Scoutmanager.CloseScoutsDataLink;
  end;
end.
```

APPENDIX B. UNIT INTERMANAGER

```
unit InterManager;

interface
{-------------------------------------------------------------------------------}

uses
CRT;

{TARGETS CONSTANTS, TYPES AND VARIABLES}
{-------------------------------------------------------------------------------}

const
NUMBEROFTGTS     =500;                                          {NUMBER OF TARGETS}
BIGNUMBER        =1E30;                                         {UTILITY VARIABLE}
DEFMOTIONMOD     ='OMNI';                                {OMNIDIRECTIONAL MOTION MODEL}
DEFSIGMARINIT    =20;                                               {DATUM ERROR}
DEFPORTMOSTCOURSE=0;                                            {LEFT LIMIT COURSE}
DEFCOURSERANGE   =360;                                             {COURSE RANGE}
DEFMINSUBSPEED   =3;                                               {MIN SUB SPEED}
DEFMAXSUBSPEED   =7;                                               {MAX SUB SPEED}
DEFREACTRANAGE   =40;                                           {REACTION RANGE}
DEFREACTIONTACK  =150;                                  {REACTION CHANGE IN COURSE}
OMNI             ='OMNI';                                          {MOTION MODEL}

type
targetype          = record                                     {TARGET STRUCTURE}
                     x              :integer;                          {X POSITION}
                     y              :integer;                          {Y POSITION}
                     course       :real;                            {TARGET COURSE}
                     speed        :real;                            {TARGET SPEED}
                     cellx        :integer;                       {TARGET CELL IN X}
       celly        :integer;                                     {TARGET CELL IN Y}
       prob         :real;                                {TARGET INDIVIDUAL WEIGHT}
       threatdis    :real;                              {THREAT DISTANCE ACCUMULATOR}
       evasioncourse:real;                                          {TACK ANGLE}
       datumx       :integer;                        {TARGET MIND DATUM POSITION IN X}
       datumy       :integer;                        {TARGET MIND DATUM POSITION IN Y}
             end;
tgtbundletype   = array[1..NUMBEROFTGTS] of targetype;             {SET OF ALL TARGETS}

var
mytarget        :tgtbundletype;
sigmarinit      :byte;                                  {STANDARD DEVIATION OF THE DATUM}
minsubspeed     :byte;                                      {ESTIMATED MIN TARGET SPEED}
maxsubspeed     :byte;                                      {ESTIMATED MAX TARGET SPEED}
courserange     :integer;                    {ESTIMATED RANGE OF COURSES FOR THE TARGET}
portmostcourse  :integer;                              {PORT LIMIT IN THE COURSE RANGE}
motionmodel     :string;                                        {TYPE OF MOTION MODEL}
reactrange      :integer;                             {DISTANCE OF REACTION FROM SEARCHER}
reactiontack    :integer;              {ANGLE OF EVASION FROM THE BEARING TO THE SEARCHER}
totprob         :real;                    {ACCUMULATOR OF TARGETS WEIGHTS AFTER UPDATING}
```

```
{SEARCHERS CONSTANTS, TYPES AND VARIABLES}
{-------------------------------------------------------------------------------}
const
MAXSCOUT    =15;                                              {MAX NUMBER OF SCOUTS}
ANGLE       =360;                                                {UTILITY CONSTANT}
NUMOFPULSES =10;                                  {NUMBER OF SONAR PULSES PER SCAN PERIOD}

{DEFAULT VARIABLE VALUES}
DEFSL       =200;                                                    {SOURCE LEVEL}
DEFNL       =80;                                                      {NOISE LEVEL}
DEFDT       =20;                                                {DETECTION THRESHOLD}
DEFDI       =20;                                                  {DIRECTIVITY INDEX}
DEFABSCOEFF =0.004;                                            {ABSORPTION COEFFICIENT}
DEFREVRBAREA=8;                                                 {REVERBERATION AREA}
DEFSCATTSTRG=-40;                                             {AREA SCATTERING STRENGTH}
DEFCONT_AOU =40;                                  {AREA OF UNCERTAINTY IN CONTACT REPORTS}
DEFFREQUENCY=6;                                                      {FREQUENCY}
DEFPULSELENG=0.05;                                                  {PULSE LENGTH}
DEFBEAMWIDTH=14;                                                     {BEAMWIDTH}
DEFNOISESIG =30;                                    {UNCERTAINTY IN SONAR EQN. TERMS}

type
scouttype       = record                                        {SEARCHERS DATA STRUCTURE}
                      x         :integer;                          {SEARCHER X POSITION}
                      y         :integer;                          {SEARCHER Y POSITION}
                      course    :integer;                             {SEARCHER COURSE}
                      speed     :real;                                {SEARCHER SPEED}
           SONAR_ON   :boolean;                                {SONAR CONDITION ON OFF}
           maxscale   :integer;          {SONAR SCALE SELECTED IN THE SEARCHER'S SONAR}
           IS_HOT     :boolean;           {WHETHER THE SEARCHER HOLDS A CONTACT OR NOT}
           abscoeff   :real;                                 {ABSORPTION COEFFICIENT}
           frequency  :real;                                        {SONAR FREQUENCY}
           pulselength:real;                                           {PULSE LENGTH}
           reverbarea :integer;     {AREA OF REVERBERATION FOR GIVEN LENGTH AND BEAMWIDTH}
           beamwidth  :real;                                        {SONAR BEAMWIDTH}
           SL         :integer;                                        {SOURCE LEVEL}
           NL         :byte;          {NOISE LEVEL CORRESPONDING TO THE SEARCHER SPEED}
           DT         :byte;                                 {DETECTION THRESHOLD}
           DI         :byte;                                   {DIRECTIVITY INDEX}
           SS         :integer;    {SCATTERING STRENGTH CORRESPONDING TO SONAR CONDITIONS}
           subx       :integer;                          {X POSITION OF A CONTACT}
           suby       :integer;                          {Y POSITION OF A CONTACT}
           subcourse  :integer;                                {CONTACT COURSE}
           subspeed   :integer;                                 {CONTACT SPEED}
           cred       :real;                        {CREDIBILITY IN THE CONTACT REPORT}
           id         :char;                      {PLATFORM IDENTITY, SURFACE AIR BUOY}
           initialhot :integer;                          {TIME OF INITIAL CONTACT}
                end;
scoutteamtype   = array[1..MAXSCOUT] of scouttype;        {STRUCTURE TYPE OF ALL SEARCHERS}

var
myscout         :scoutteamtype;                                 {INSTANCIATED SEARCHERS}
numberofscouts  :byte;                                {TOTAL NUMBER OF INSTANCIATED SCOUTS}
noisesigma      :byte;          {STANDARD DEVIATION IN THE SOLUTION OF SONAR EQQUATION}
undetectedmass  :real;                                  {UNDETECTED PROBABILITY MASS}
```

```
{ENVIRONMENTAL VARIABLES AND CONSTANTS}
{-------------------------------------------------------------------------------}
const
DEFBOTTOMTYP=1;                                              {BOTTOM TYPE}
DEFSVTPROFIL=1;                                    {SOUND VELOCITY PROFILE TYPE}
DEFWIND     =10;                                      {WIND INTENSITY IN KNOTS}
DEFDEPTH    =200;                                                {DEPTH IN M}

var
bottomtype     :byte;                                {TYPE OF BOTTOM I II OR III}
svtprofiletype :byte;                          {TYPE OF VERTICAL SOUND SPEED PROFILE}
wind           :integer;                             {WIND INTENSITY IN KNOTS}
depth          :integer;                                          {DEPTH IN M}

{SIMULATION AND CONTROL FLOW PARAMETERS}
{-------------------------------------------------------------------------------}
const
DEFTOTALSIMTIME = 15;                                        {TOTAL SIMULATION TIME}
DEFSCOUTSDELAY  = 6;                                           {SCOUTS DATUM DELAY}

var
WannaContinue :boolean;                                       {EXIT BOOLEAN CONDITION}
WannaSeeData  :boolean;                                {GRAPHIC DISPLAY BOOLEAN CONDITON}
datalink      :text;                                {LOCATION OF THE DATA LINK DEVICE DRIVER}
dlinktotaltime :integer;                {TOTAL SIMULATION TIME IN DATA LINK DEVICE DRIVER}
totalsimtime   :word;                     {TIME IN MINUTES FOR WHICH A RENDER IS DESIRED}
time           :word;                                    {DISCRETE TIME UNITS COUNTER}
scoutsdelay    :real;                                     {TIME OFF OF THE SEARCHERS}



{PUBLIC METHODS}
{-------------------------------------------------------------------------------}

procedure GetTargetData;
procedure GetScoutsData;
procedure GetEnvironmentalData;
procedure GetSimData;
procedure AskOptions;
```

```
implementation
{══════════════════════════════════════════════════════════════════════════════}

uses detecmanager;

procedure PromptOptions;
var i :integer;
begin
clrscr;
textcolor(Cyan);
clrscr;
writeln('     ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('     ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒NRT   CAS▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('     ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('     Please, select one of the following options:');
writeln;
for i:=1 to 16 do writeln;
textcolor(Red);
writeln('                           0. QUIT PROGRAM');
writeln('                           1. UPDATE TARGET DATA');
writeln('                           2. UPDATE SEARCHER DATA');
writeln('                           3. UPDATE ENVIRONMENTAL DATA' );
writeln('                           4. SET TIME VARIABLES');
writeln('                           5. START SIMULATION WITH CURRENT DATA');
writeln('                           6. DISPLAY INITIAL TARGET POSITIONS' );
writeln;
write  ('                              Enter your option: ');
end;


procedure AskOptions;
var
answer          :char;
origmode        :integer;
begin
answer:='9';
origmode:=LastMode;
TextMode(C80+Font8x8);
while ((answer<>'5') and (answer<>'6')) and(answer<>'0') do
      begin
      PromptOptions;
      answer:=readkey;
      case answer of
          '0': WannaContinue:=FALSE;
          '1': GetTargetData;
          '2': GetScoutsData;
          '3': GetEnvironmentalData;
          '4': GetSimData;
          '5': ;
          '6': WannaSeeData:=TRUE;
      end;
end;
TextMode(origmode);
end;
```

```
procedure GetTargetReaction;
var
count :byte;
begin
textcolor(Red);
clrscr;
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒       TARGET DATA     ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    Please input the following data for 500 targets:');
for count:=1 to 12 do writeln;
writeln;writeln;
writeln('    TARGET REACTION PARAMETERS                                     ');
writeln;
writeln;
writeln;
writeln('    Please input the estimated REACTION RANGE in nautical miles,    ');
writeln;
writeln('    The current value is                            :  ',reactrange/10:3:1);
write  ('    Enter your value                                :     ');
readln(reactrange);
reactrange:=reactrange*10;
writeln;
writeln('    Please input the estimated REACTION TACK in nautical degrees,    ');
writeln('    (Evasion course relative to the bearing to the closest scout     ');
writeln;
writeln('    The current value is                            :  ',reactiontack:3);
write  ('    Enter your value                                :     ');
readln(reactiontack);
end;


procedure GetDatumError;
var
count :byte;
begin
textcolor(Red);
clrscr;
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒       TARGET DATA     ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    Please input the following data for 500 targets:');
for count:=1 to 12 do writeln;
writeln;writeln;
writeln('    DATUM OMNIDIRECTIONAL ERROR (integer)                                     ');
writeln('    (Standard Deviation in Tgts Position, in nm                    ');
writeln;
writeln('    Current value is                                :  ',sigmarinit/10:3:0);
write  ('    Enter your value                                :     ');
readln(sigmarinit);
sigmarinit:=sigmarinit*10;
writeln;
writeln;
writeln;
writeln;
end;
```

```
procedure GetMotionModel;
var
ans   :byte;
count :byte;
begin
textcolor(Red);
clrscr;
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒       TARGET DATA        ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    Please input the following data for 500 targets:');
for count:=1 to 12 do writeln;
writeln;writeln;
writeln('    TARGET MOTION MODEL                                  ');
writeln;
writeln('            (1) OMNI : Omnidirectional fleeing                                ');
writeln('            (2) FAN  : Directional movement');
writeln;
writeln('    Current motion model is                      :  ',motionmodel);
write  ('    Enter your model (1 OMNI, 2 FAN)             :  ');
readln(ans);
case ans of
   1: begin
        motionmodel := 'OMNI';
        portmostcourse:=0;
        courserange:=360;
      end;
   2: motionmodel := 'FAN';
end;
end;


procedure GetTargetCourses;
var
count :byte;
begin
textcolor(Red);
clrscr;
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒       TARGET DATA        ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    Please input the following data for 500 targets:');
for count:=1 to 12 do writeln;
writeln;writeln;
writeln('    TARGET COURSE LIMITS                                       ');
writeln;
writeln;
writeln;
writeln('    Please input the courses port limit                        ');
writeln;
writeln('    The current value is                         :  ',portmostcourse:3);
write  ('    Enter your value                             :    ');
readln(portmostcourse);
writeln;
writeln('    Please input the course range                              ');
writeln;
writeln('    The current value is                         :  ',courserange:3);
write  ('    Enter your value                             :    ');
readln(courserange);
writeln;
end;
```

```
procedure GetTargetSpeeds;
var
count :byte;
begin
textcolor(Red);
clrscr;
writeln('    ░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░');
writeln('    ░░░░░░░░░░░░░░░░░░░░░░       TARGET DATA            ░░░░░░░░░░░░░░░░░░');
writeln('    ░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░');
writeln('    Please input the following data for 500 targets:');
for count:=1 to 12 do writeln;
writeln;writeln;
writeln('    TARGET SPEED LIMITS                                        ');
writeln;
writeln;
writeln;
writeln('    Please input the Minimum estimated Target speeed           ');
writeln;
writeln('    The current value is                             :  ',round(minsubspeed*1.2):3);
write  ('    Enter your value                                 :     ');
readln(minsubspeed);
minsubspeed:= round(0.8333*minsubspeed);{converting to discrete speed units}
writeln;
writeln('    Please input the Maximum estimated Target speed            ');
writeln;
writeln('    The current value is                             :  ',round(maxsubspeed*1.2):3);
write  ('    Enter your value                                 :     ');
readln(maxsubspeed);
maxsubspeed:= round(0.8333*maxsubspeed);{converting to discrete speed units}
writeln;
end;


procedure GetTargetData;
var
answering   : char;
count       : byte;
begin
answering:='P';
while answering <> '0' do begin
textcolor(Red);
clrscr;
writeln('    ░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░');
writeln('    ░░░░░░░░░░░░░░░░░░░░░░       TARGET DATA            ░░░░░░░░░░░░░░░░░░');
writeln('    ░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░');
writeln('    Default values of the parameters used to generate and update ');
writeln('    500 simulated tracks: ');
for count:=1 to 8 do writeln;
writeln;
writeln('       (0)  Back to Main                                          ');
writeln;
write  ('       (1)  DATUM ERROR IN NM                         :  ',sigmarinit/10:3:0);
writeln;writeln;
writeln('       (2)  TARGET MOTION MODEL                       :  ',motionmodel);
writeln('            (speed & course uniformly distributed)');
writeln;
writeln('       (3)  COURSE LIMITS (FOR FAN MOTION MODEL ONLY)       ');
writeln;
writeln('            PORT LIMIT (0-360)                        :  ',portmostcourse:3);
writeln('            RANGE      (Not starbord limit)           :  ',courserange:3);
writeln;
writeln('       (4)  SPEED');
```

```
writeln;
writeln('           MINIMUM ESTIMATED TARGET SPEED IN KTS   :  ',round(minsubspeed*1.2):3);
writeln('           MAXIMUM ESTIMATED TARGET SPEED IN KTS   :  ',round(maxsubspeed*1.2):3);
writeln;
writeln('      (5)  TARGET REACTION MODEL                       ');
writeln;
writeln('           REACTION RANGE                          :  ',reactrange/10:3:0);
writeln('           REACTION TACK                           :  ',reactiontack:3);
writeln;
writeln('      Enter the number of the parameters you want to change or 0 ');
write  ('      to go back to main. Return is not necessary  :  ' );
answering:=readkey;
case answering of
    '1'  : GetDatumError;
    '2'  : GetMotionModel;
    '3'  : GetTargetCourses;
    '4'  : GetTargetSpeeds;
    '5'  : GetTargetReaction;
end;
end;{while}
end;{GetTargetData}



procedure DisplayDataLinkInfo;
var
k        :byte;
begin
clrscr;
writeln('    ░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░');
writeln('    ░░░░░░░░░░░░░░░░░░░░░░░    SCOUTS   DATA          ░░░░░░░░░░░░░░░░░░');
writeln('    ░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░');

write  ('    Connected to Data Link, # of scouts present    : ');
textcolor(140);
writeln('    --> ',numberofscouts);
textcolor(red);
write  ('                            Time searching so far  : ');
textcolor(140);
writeln('    --> ',dlinktotaltime*5,' min');
textcolor(red);
writeln;
writeln('    Receiving the following initial conditions:');
writeln;
writeln;
writeln('     SCOUT #    XPOS   YPOS    COURSE    SPEED   SONAR_ON   SCALE        ');
writeln;
for k:=1 to numberofscouts do begin
   with myscout[k] do begin
      writeln('    ',k:8 ,' ', x:8  , y:8,  course:8, speed:8:0,SONAR_ON:8,maxscale*185:8);
   end;
end;
writeln;
writeln;
writeln('      Hit any key to continue...');
readkey;
end;
```

```
procedure ChangeSonarDefaults(scout,param:byte);
var
f        :real;
k        :byte;
begin
clrscr;
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒   SCOUTS   DATA        ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    Scouts Sonar Parameters:');
writeln;
writeln;
writeln('    ┌──────────────────────────────────────────────────────────────┐ ');
writeln('    │         (1)      (2)      (3)      (4)      (5)         (6)      │ ');
writeln('    │SCOUT    SL       DT       DI       Frq.     PulseL.     Beam W.  │ ');
writeln('    │  #      (dB)     (dB)     (dB)     (Khz)    (msec)      (deg)    │ ');
writeln('    ├──────────────────────────────────────────────────────────────┤ ');
for k:=1 to numberofscouts do begin
   with myscout[k] do begin
      writeln('    │',k:5  ,'   ',SL:5,'   ' ,DT:5,'   ',DI:5,' ',frequency:8:1,'  '
              ,pulselength*1000:8:0,'  ',beamwidth:8:1,'    │');
   end;
end;
writeln('    └──────────────────────────────────────────────────────────────┘');
writeln;
writeln('    You are going to produce modifications to scout number: ' ,scout);
with myscout[scout] do begin
   if param = 1 then begin
      writeln;
      write('    Please input new SOURCE LEVEL                              : ');
      readln(SL);
   end;
   if param = 2 then begin
      writeln;
      write('    Please input new DETECTION THRESHOLD                       : ');
      readln(DT);
   end;
   if param = 3 then begin
      writeln;
      write('    Please input new DIRECTIVITY INDEX                         : ');
      readln(DI);
   end;
   if param = 4 then begin
      writeln;
      write('    Please input new FREQUENCY(updates aborption coeff)   : ');
      readln(frequency);
      abscoeff:=AbsorptionCoefficient(frequency);
   end;
   if param = 5 then begin
      writeln;
      write('    Please input new PULSE LENGTH(in ms)                       : ');
      readln(pulselength);
      pulselength:=pulselength/1000;
   end;
   if param = 6 then begin
      writeln;
      write('    Please input new BEAM WIDTH(in degrees)                     : ');
      readln(beamwidth);
   end;
end;
end;
```

```
procedure DisplayandChangeSonarDefaults;
var
k,item   :byte;
scoutmod :byte;
option   :char;
begin
while scoutmod <> 0 do begin
  clrscr;
  writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
  writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒   SCOUTS   DATA         ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
  writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
  writeln('    Scouts Sonar Parameters:');
  writeln;
  writeln;
  writeln('    ┌──────────────────────────────────────────────────────────────────┐ ');
  writeln('    │        (1)      (2)      (3)      (4)      (5)         (6)      │ ');
  writeln('    │SCOUT   SL       DT       DI       Frq.     PulseL.     Beam W.  │ ');
  writeln('    │  #     (dB)     (dB)     (dB)     (Khz)    (msec)      (deg)    │ ');
  writeln('    ├──────────────────────────────────────────────────────────────────┤ ');
  for k:=1 to numberofscouts do begin
    with myscout[k] do begin
      writeln('    │',k:5  ,'   ',SL:5,'   ' ,DT:5,'   ',DI:5,' ',frequency:8:1,'  '
              ,pulselength*1000:8:0,'  ',beamwidth:8:1,'    │');
    end;
  end;
  writeln('    └──────────────────────────────────────────────────────────────────┘');
  writeln;
  writeln('     Please input the number of the searcher you want to modify  ');
  write  ('     or 0 to go back to Scouts Menu  : ');
  readln(scoutmod);
  writeln;
  if scoutmod <> 0 then begin
     if scoutmod > numberofscouts then begin
        writeln('Please input scout # smaller than ',numberofscouts);
        writeln('Hit any key to continue..');
        readkey;
     end else begin
        write ('      Please input the column of the item you want to modify    :  ');
        readln(item);
        writeln;
        ChangeSonarDefaults(scoutmod,item);
     end;
  end;{if}
end;{while}
end;


procedure DisplayandChangeErrorDefaults;
var
thisanswer  :integer;
begin
  clrscr;
  writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
  writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒   SCOUTS   DATA         ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
  writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
  writeln('                        ERROR VARIABLE DEFAULT VALUE             ');
  writeln;
  writeln;
  writeln;
  writeln('    Total estimated error in the calculation of the terms in the     ');
  writeln('    calculation of the sonar equation and in the uncertainties in    ');
```

```
  writeln('    the detection process in dB re µPascal,
    ');
  writeln;
  writeln('    "NOISESIGMA"  current value is        : ',noisesigma:6 ,' dB re µPascal');
  writeln;
  write  ('    Please input the new valaue           : ');
  readln(noisesigma);
end; {procedure}



procedure GetScoutsData;
var
response      :char;
begin
response:='9';
while response <> '0'do begin
  clrscr;
  writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
  writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒    SCOUTS   DATA        ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
  writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
  writeln('                              ');
  writeln('    Please select one of the following options:                   ');
  writeln;
  writeln;
  writeln;
  writeln;
  writeln;
  writeln('                     0.  BACK TO MAIN                        ');
  writeln('                     1.  DISPLAY INFO RECEIVED FROM SCOUTS      ');
  writeln('                     2.  DISPLAY AND OR CHANGE DEFAULT SONAR PARAMETERS');
  writeln('                     3.  DISPLAY AND OR CHANGE DEFAULT ERROR VALUES  ');
  writeln;
  write  ('                         Enter your option:  ');
  response:=readkey;
  case response of
    '1': DisplayDataLinkInfo;
    '2': DisplayandChangeSonarDefaults;
    '3': DisplayandChangeErrorDefaults;
    '0': ;
  end;
end;
end;



procedure GetSimData;
var
answer:char;
begin
clrscr;
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒      SIMULATION    TIME        ▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    The simulation will be carried out from time 0 to TotalSimtime  ');
writeln('    in updating steps of 5 minutes. Totalsimtime should be smaller  ');
writeln('    than the last updating time to the data-link device file.       ');
writeln('                                                                     ');
writeln;
```

```
writeln;
writeln;
writeln('    Total Simulation Time (TotalSimtime, a multiple of 5 min)');
write  ('    No info from scouts available after          :  ');
textcolor(140);
writeln(dlinktotaltime*5,'min');
textcolor(red);
writeln;
writeln('    Current value is                             :  ',totalsimtime*5,' min');
write  ('    Enter your value                             :  ');
readln(totalsimtime);
totalsimtime:=totalsimtime div 5;
if totalsimtime > dlinktotaltime then begin
writeln('    Simulation time should be smaller than ',dlinktotaltime*5);
   writeln;writeln;writeln;writeln;writeln;
   writeln('     Hit any key to continue...');
   readkey;
end;
writeln;
writeln;
writeln('    Delay Time of scouts after DATUM time,(DATUM "age" in min) ');
writeln('    Current value is                             :  ',scoutsdelay*5:4:0);
write  ('    Enter your value                             :  ');
readln(scoutsdelay);
scoutsdelay:=scoutsdelay/5;
end;


function Profileinwords:string;
var
a:string;
begin
   case svtprofiletype of
      1:  a:='POSITIVE OR ISOSPEED';
      2:  a:='NEGATIVE';
   end;
Profileinwords:=a;
end;

function Bottominwords:string;
begin
   case bottomtype of
      1:  Bottominwords:='SANDY, FLAT';
      2:  Bottominwords:='INTERMEDIATE BETWEEN VERY ROUGH AND FLAT';
      3:  Bottominwords:='ROCKY, VERY ROUGH';
   end;
end;

procedure Getwindspeed;
begin
clrscr;
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒      ENVIRONMENTAL   DATA        ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    This is the current environmental data:                          ');
writeln;
writeln;
writeln;
writeln('          (1) WINDSPEED            : ',wind:4,' Knots'          );
writeln('          (2) SOUND SPEED PROOFILE :   ',Profileinwords         );
writeln('          (3) BOTTOM TYPE          :   ',Bottominwords          );
writeln('          (4) DEPTH                : ',depth:4,' m'             );
```

```
writeln;
write  ('    Please input wind speed (knots):   '                        );
readln(wind);
end;

procedure GetSoundProfile;
var
OUTTAHERE  :boolean;
begin
OUTTAHERE:=FALSE;
while not OUTTAHERE do begin
clrscr;
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒      ENVIRONMENTAL   DATA       ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    This is the current environmental data:                       ');
writeln;
writeln;
writeln('          (1) WINDSPEED            : ',wind:4,' Knots'            );
writeln('          (2) SOUND SPEED PROOFILE :   ',Profileinwords           );
writeln('          (3) BOTTOM TYPE          :   ',Bottominwords            );
writeln('          (4) DEPTH                : ',depth:4,' m'               );
writeln;
writeln;
writeln('    You can choose the sound speed profile according to the following');
writeln('    code:');
writeln;
writeln('          (1) POSITIVE OR ISOSPEED                                ');
writeln('          (2) NEGATIVE                                            ');
writeln;
write  ('    Please enter your option here  :   '                          );
readln(svtprofiletype);
if (svtprofiletype= 1) or (svtprofiletype = 2) then begin
  OUTTAHERE:=TRUE;
end;
end;

procedure GetBottomtype;
var
OUTTAHERE  :boolean;
begin
OUTTAHERE:=FALSE;
while not OUTTAHERE do begin
clrscr;
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒      ENVIRONMENTAL   DATA       ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒');
writeln('    This is the current bottom & depth environmental data:
');
writeln;
writeln;
writeln('          (1) WINDSPEED            : ',wind:4,' Knots'            );
writeln('          (2) SOUND SPEED PROOFILE :   ',Profileinwords           );
writeln('          (3) BOTTOM TYPE          :   ',Bottominwords            );
writeln('          (4) DEPTH                : ',depth:4,' m'               );
writeln;
writeln;
writeln('    You can input bottom type according to the following code');
writeln;
writeln('          (1) SANDY,FLAT BOTTOM,NEGLECTABLE ROUGHNEESS            ');
writeln('          (2) INTERMEDIATE BETWEEN FLAT AND VERY ROUGH            ');
writeln('          (3) ROCKY, VERY ROUGH                                   ');
```

```
writeln;
write  ('    Please enter your option here  :   '                           );
readln(bottomtype);
if (bottomtype = 1) or (bottomtype = 2) or (bottomtype = 3) then begin
  OUTTAHERE:=TRUE;
end;
end;
end;



procedure Getdepth;
begin
clrscr;
writeln('    ░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░');
writeln('    ░░░░░░░░░░░░░░░      ENVIRONMENTAL   DATA          ░░░░░░░░░░░░░');
writeln('    ░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░');
writeln('    This is the current environmental data:                        ');
writeln;
writeln;
writeln;
writeln('          (1) WINDSPEED            : ',wind:4,' Knots'          );
writeln('          (2) SOUND SPEED PROOFILE :   ',Profileinwords         );
writeln('          (3) BOTTOM TYPE          :   ',Bottominwords          );
writeln('          (4) DEPTH                : ',depth:4,' m'             );
writeln;
write  ('    Please input depth (meters):   '                           );
readln(depth);
end;



procedure GetEnvironmentalData;
var
hisanswer   : char;
begin
while hisanswer <> '0' do begin
clrscr;
writeln('    ░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░');
writeln('    ░░░░░░░░░░░░░░░      ENVIRONMENTAL   DATA          ░░░░░░░░░░░░░');
writeln('    ░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░');
writeln('    This is the current environmental data:                        ');
writeln;
writeln;
writeln;
writeln('          (0) BACK TO MAIN                                      ');
writeln('          (1) WINDSPEED            : ',wind:4,' Knots'          );
writeln('          (2) SOUND SPEED PROOFILE :   ',Profileinwords         );
writeln('          (3) BOTTOM TYPE          :   ',Bottominwords          );
writeln('          (4) DEPTH                : ',depth:4,' m'             );
writeln;
writeln('    Please input the numer of the                               ');
write  ('    parameters you want to change   :                            ');
hisanswer:=readkey;
case hisanswer of
  '1'     :    Getwindspeed;
  '2'     :    GetSoundProfile;
  '3'     :    GetBottomtype;
  '4'     :    GetDepth;
end;
end;
end;
```

```
begin

{TARGET INIT}
sigmarinit     :=DEFSIGMARINIT;
portmostcourse :=DEFPORTMOSTCOURSE;
courserange    :=DEFCOURSERANGE;
minsubspeed    :=DEFMINSUBSPEED;
maxsubspeed    :=DEFMAXSUBSPEED;
reactrange     :=DEFREACTRANAGE;
reactiontack   :=DEFREACTIONTACK;
motionmodel    :=OMNI;


{SEARCHER INIT}
noisesigma:=DEFNOISESIG;

{SIMULATION INIT}

totalsimtime :=DEFTOTALSIMTIME;
WannaContinue:=TRUE;
scoutsdelay  :=DEFSCOUTSDELAY;


{ENVIRONMENT INIT}
bottomtype     :=DEFBOTTOMTYP;
svtprofiletype :=DEFSVTPROFIL;
wind           :=DEFWIND;
depth          :=DEFDEPTH;
end.
```

APPENDIX C. USER INTERFACE SEQUENCES

1.0 GENERAL MENU

```
                    NRT   CAS

Please, select one of the following options:


                    0. QUIT PROGRAM
                    1. UPDATE TARGET DATA
                    2. UPDATE SEARCHER DATA
                    3. UPDATE ENVIRONMENTAL DATA'
                    4. SET TIME VARIABLES
                    5. START SIMULATION WITH CURRENT DATA
                    6. DISPLAY INITIAL TARGET POSITIONS'

                      Enter your option:
```

2.1 TARGET PARAMETERS MAIN MENU

```
                        TARGET DATA

Default values of the parameters used to generate and update
500 simulated tracks:


   (0)  Back to Main

   (1)  DATUM ERROR IN NM                          :

   (2)  TARGET MOTION MODEL                        :
        (speed & course uniformly distributed)

   (3)  COURSE LIMITS (FOR FAN MOTION MODEL ONLY)

        PORT LIMIT (0-360)                         :
        RANGE      (Not starbord limit)            :

   (4)  SPEED

        MINIMUM ESTIMATED TARGET SPEED IN KTS      :
        MAXIMUM ESTIMATED TARGET SPEED IN KTS      :

   (5)  TARGET REACTION MODEL

        REACTION RANGE                             :
        REACTION TACK                              :

   Enter the number of the parameters you want to change or 0
   to go back to main. Return is not necessary  :  '
```

2.2 ERROR INPUT

```
                    TARGET DATA
Please input the following data for 500 targets:


DATUM OMNIDIRECTIONAL ERROR (integer)
(Standard Deviation in Tgts Position, in nm

Current value is                          :
Enter your value                          :
```

2.3. MOTION MODEL INPUT

```
                    TARGET DATA
Please input the following data for 500 targets:

TARGET MOTION MODEL

        (1) OMNI : Omnidirectional fleeing
        (2) FAN  : Directional movement

Current motion model is                   :
Enter your model (1 OMNI, 2 FAN)          :
```

2.4. TARGET COURSE LIMITS INPUT

```
                    TARGET DATA
Please input the following data for 500 targets:


TARGET COURSE LIMITS

Please input the courses port limit

The current value is                      :
Enter your value                          :

Please input the course range

The current value is                      :
Enter your value                          :
```

2.5. TARGET SPEED ESTIMATE INPUTS

```
                    TARGET DATA
Please input the following data for 500 targets:


TARGET SPEED LIMITS

Please input the Minimum estimated Target speeed

The current value is                    :
Enter your value                        :


Please input the Maximum estimated Target speed

The current value is                    :
Enter your value                        :
```

2.6. TARGET REACTION PARAMETERS INPUTS

```
                    TARGET DATA
Please input the following data for 500 targets:

TARGET REACTION PARAMETERS

Please input the estimated REACTION RANGE in nautical miles,

The current value is                    :
Enter your value                        :


Please input the estimated REACTION TACK in nautical degrees,
(Evasion course relative to the bearing to the closest scout

The current value is                    :
Enter your value                        :
```

3.1 SCOUTS MAIN MENU

```
                    SCOUTS   DATA

Please select one of the following options:

          0.  BACK TO MAIN
          1.  DISPLAY INFO RECEIVED FROM SCOUTS
          2.  DISPLAY AND OR CHANGE DEFAULT SONAR PARAMETERS
          3.  DISPLAY AND OR CHANGE DEFAULT ERROR VALUES

                Enter your option:
```

3.2. SCOUTS DATA LINK DISPLAY

```
                    SCOUTS   DATA

Connected to Data Link, # of scouts present   :

                       Time searching so far :

Receiving the following initial conditions    :

| SCOUT # | XPOS | YPOS | COURSE | SPEED | SONAR_ON | SCALE |
|---------|------|------|--------|-------|----------|-------|
|         |      |      |        |       |          |       |
|         |      |      |        |       |          |       |
|         |      |      |        |       |          |       |
|         |      |      |        |       |          |       |

Hit any key to continue...
```

3.3. SONAR PARAMETERS DISPLAY AND INPUT

SCOUTS DATA

Scouts Sonar Parameters:

SCOUT #	(1) SL (dB)	(2) DT (dB)	(3) DI (dB)	(4) Frq. (Khz)	(5) PulseL. (msec)	(6) Beam W. (deg)

You are going to produce modifications to scout number:

(eventual)

Please input new SOURCE LEVEL :

Please input new DETECTION THRESHOLD :

Please input new DIRECTIVITY INDEX :

Please input new FREQUENCY(updates aborption coeff) :

Please input new PULSE LENGTH(in ms) :

Please input new BEAM WIDTH(in degrees) :

3.4. SONAR EQUATION ERROR INPUT

SCOUTS DATA

ERROR VARIABLE DEFAULT VALUE

Total estimated error in the calculation of the terms in the calculation of the sonar equation and in the uncertainties in the detection process in dB re μPascal,

"NOISESIGMA" current value is :

Please input the new valaue :

4.1 ENVIRONMENTAL MAIN MENU

```
                    ENVIRONMENTAL   DATA
This is the current environmental data:


      (0) BACK TO MAIN
      (1) WINDSPEED            :
      (2) SOUND SPEED PROOFILE :
      (3) BOTTOM TYPE          :
```

4.2. WIND SPEED INPUT

```
                    ENVIRONMENTAL   DATA
This is the current environmental data:


      (1) WINDSPEED            :
      (2) SOUND SPEED PROOFILE :
      (3) BOTTOM TYPE          :
      (4) DEPTH                :

   Please input wind speed (knots):
```

4.3. SOUND SPEED PROFILE TYPE INPUT

```
                    ENVIRONMENTAL   DATA
This is the current environmental data:


      (1) WINDSPEED            :
      (2) SOUND SPEED PROOFILE :
      (3) BOTTOM TYPE          :
      (4) DEPTH                :


You can choose the sound speed profile according to the following
code:

      (1) POSITIVE OR ISOSPEED
      (2) NEGATIVE

   Please enter your option here  :
```

4.4. BOTTOM TYPE INPUT

```
                    ENVIRONMENTAL   DATA
This is the current bottom & depth environmental data:


      (1) WINDSPEED            :
      (2) SOUND SPEED PROOFILE :
      (3) BOTTOM TYPE          :
      (4) DEPTH                :


You can input bottom type according to the following code

      (1) SANDY,FLAT BOTTOM,NEGLECTABLE ROUGHNEESS
      (2) INTERMEDIATE BETWEEN FLAT AND VERY ROUGH
      (3) ROCKY, VERY ROUGH

   Please enter your option here  :   '
```

4.5. DEPTH INPUT

```
                    ENVIRONMENTAL   DATA
This is the current environmental data:


      (1) WINDSPEED            :
      (2) SOUND SPEED PROOFILE :
      (3) BOTTOM TYPE          :
      (4) DEPTH                :

   Please input depth (meters):   '
```

5.0 TIME VARIABLES INPUT

```
                    SIMULATION    TIME
The simulation will be carried out from time 0 to TotalSimtime
in updating steps of 5 minutes. Totalsimtime should be smaller
than the last updating time to the data-link device file.



Total Simulation Time (TotalSimtime, a multiple of 5 min)
No info from scouts available after          :

Current value is                             :
Enter your value                             :


Hit any key to continue...



Delay Time of scouts after DATUM time,(DATUM "age" in min)
Current value is                             :
Enter your value                             :
```

APPENDIX D. TARGETMANAGER UNIT CODE

```
unit TargtManager;

interface

uses
GRAPH, CRT;
var

function  normal(var u:double):real;
procedure GenTargets;
procedure TargetReaction(ti,ss:integer;diststt:real);
procedure UpdateTargetPosition(ti:integer);
procedure UpdateWeightsWithBayes;


implementation

uses GraphManager,InterManager;


function normal(var u:double):real;
{converts uniform random numbers to standard normal}
{by Alan Washburn}
const
   a=2.515517;b=0.802853;c=0.010328;d=1.432788;e=0.189269;f=0.001308;
var
   sign:integer;
   x,y:real;
begin
   sign:=-1;
   if u>0.5 then begin
      sign:=1;
      u:=1.0-u;
   end;
   y:=sqrt(-ln(u*u));
   x:=(a+y*(b+c*y))/(1+y*(d+y*(e+f*y)));
   normal:=sign*(y-x);
end;{normal}


function bearing(x1,y1,x2,y2:integer):integer;
{FROM  1 TO 2}
var
x,y   :integer;

begin
x:=x2-x1;
y:=y2-y1;

if x>0 then begin
   if y > 0 then bearing := round(180 - arctan(x/y)*180/pi);
   if y = 0 then bearing := 90;
   if y < 0 then bearing := -round(arctan(x/y)*180/pi);
end;
if x<0 then begin
   if y < 0 then bearing := round(360 - arctan(x/y)*180/pi);
   if y = 0 then bearing := 270;
```

```
   if y > 0 then bearing := round(180 - arctan(x/y)*180/pi);
end;
if x=0 then begin
   if y >=0 then bearing := 180;
   if y < 0 then bearing := 0;
end;
end;


procedure GenTargets;
var
r           :double;
i           :word;
datumerror  :integer;
range       :real;
begin
   Randomize;
   for i:=1 to NUMBEROFTGTS do begin
      r:=random;
      mytarget[i].x := CENTER + round(normal(r) * sigmarinit);
      r:=random;
      mytarget[i].y := CENTER + round(normal(r) * sigmarinit);
      mytarget[i].datumx:=mytarget[i].x;
      mytarget[i].datumy:=mytarget[i].y;
      mytarget[i].course:=portmostcourse+random(courserange);
      mytarget[i].speed:=minsubspeed+random*(maxsubspeed-minsubspeed);
      mytarget[i].x:=mytarget[i].x + round(mytarget[i].speed * scoutsdelay *
                                    sin(mytarget[i].course*pi/180));
      mytarget[i].y:=mytarget[i].y - round(mytarget[i].speed * scoutsdelay *
                                    cos(mytarget[i].course*pi/180));
      mytarget[i].cellx :=(mytarget[i].x div GRIDSTEP)+1;
      mytarget[i].celly :=(mytarget[i].y div GRIDSTEP)+1;
      mytarget[i].prob:=0.002;
      mytarget[i].threatdis:=BIGNUMBER;
   end;
end;


procedure UpdateTargetPosition(ti:integer);
var
r              :double;
begin
   mytarget[ti].x:=mytarget[ti].x + round(mytarget[ti].speed *
                                    sin(mytarget[ti].course*pi/180)) ;
   mytarget[ti].y:=mytarget[ti].y - round(mytarget[ti].speed *
                                    cos(mytarget[ti].course*pi/180));
   mytarget[ti].cellx :=(mytarget[ti].x div GRIDSTEP)+1;
   mytarget[ti].celly :=(mytarget[ti].y div GRIDSTEP)+1;
end;



procedure TargetReaction(ti,ss:integer;distst:real);
var
filehandle      :text;
r               :double;
bearts          :integer;
begin
   if distst < reactrange then begin
      bearts:=bearing(mytarget[ti].x,mytarget[ti].y,myscout[ss].x,myscout[ss].y);
      if bearts > 360 then bearts:=bearts-360;
      if bearts < 0   then bearts:=360-bearts;
```

```
      if sin((myscout[ss].course-bearts)*pi/180) > 0 then begin
          mytarget[ti].course:=bearts - reactiontack;
      end else begin
          mytarget[ti].course:=bearts + reactiontack;
      end;
   end else begin
      if motionmodel = 'OMNI' then begin
         mytarget[ti].course:=bearing(mytarget[ti].datumx,mytarget[ti].datumy,
                                      mytarget[ti].x,mytarget[ti].y);
      end else begin
         mytarget[ti].course:=portmostcourse+random(courserange);
      end;
   end;
   if mytarget[ti].course>360 then begin
      mytarget[ti].course:=mytarget[ti].course-360;
   end;
   if mytarget[ti].course<0   then begin
      mytarget[ti].course:=360+mytarget[ti].course;
   end;
   if mytarget[ti].threatdis > distst then begin
      mytarget[ti].threatdis:=distst;
      mytarget[ti].evasioncourse:=mytarget[ti].course;
   end;
   mytarget[ti].course:=mytarget[ti].evasioncourse;
   if ss=numberofscouts then mytarget[ti].threatdis:=BIGNUMBER;
end;{TargetReaction}



procedure UpdateWeightsWithBayes;
var
i     :integer;
begin
for i:=1 to NUMBEROFTGTS do mytarget[i].prob:=mytarget[i].prob/totprob;
end;



begin
end.
```

APPENDIX E. SCOUTMANAGER UNIT

```
unit ScoutManager;

interface

uses
GRAPH, CRT;

procedure InitScouts;
procedure CloseScoutsDataLink;
procedure ConnectWithScoutsThruDataLink;
procedure UpdatePriorWithScoutsInfo;
procedure ReadScoutsDataLink;
procedure ResetUndetectedMass;
procedure UpdateUndetectedMass;

implementation

uses GraphManager,TargtManager,Intermanager,DetecManager;


procedure InitializeScoutsVarsWithDefaults;
var
k        :integer;
begin
   for k:=1 to numberofscouts do begin
      myscout[k].abscoeff  :=DEFabscoeff;
      myscout[k].SL        :=DEFSL;
      myscout[k].NL        :=DEFNL;
      myscout[k].DT        :=DEFDT;
      myscout[k].DI        :=DEFDI;
      myscout[k].SS        :=DEFSCATTSTRG;
      myscout[k].IS_HOT    :=FALSE;
      myscout[k].frequency :=DEFFREQUENCY;
      myscout[k].beamwidth :=DEFBEAMWIDTH;
      myscout[k].pulselength:=DEFPULSELENG;
      myscout[k].reverbarea:=DEFREVRBAREA;
   end;
end;


procedure DetermineNumberOfScoutsAndTime;
var
i  :integer;
begin
readln(datalink,numberofscouts,dlinktotaltime);
for i:=1 to numberofscouts do begin
   readln(datalink,myscout[i].id);
end;
end;


procedure InitScouts;
begin
assign(datalink,'c:\lnk1.dat');
reset(datalink);
DetermineNumberOfScoutsAndTime;
close(datalink);
InitializeScoutsVarsWithDefaults;
end;
```

```
procedure CloseScoutsDataLink;
begin
close(datalink);
end;


procedure ConnectWithScoutsThruDataLink;
begin
reset(datalink);
DetermineNumberOfScoutsAndTime;
ReadScoutsDatalink;
end;


procedure ReadScoutsDataLink;
var
sonarinfo  :byte;
contactinfo:byte;
k          :byte;
begin
   for k:=1 to numberofscouts do begin
      read(datalink,myscout[k].x);
      read(datalink);
      read(datalink,myscout[k].y);
      read(datalink,myscout[k].course);
      read(datalink,myscout[k].speed);
      read(datalink,sonarinfo);
      read(datalink,myscout[k].maxscale);
      read(datalink,contactinfo);
      if sonarinfo = 1 then begin
         myscout[k].SONAR_ON:=TRUE;
      end;
      if contactinfo = 1 then begin
         if not myscout[k].IS_HOT then begin
            myscout[k].initialhot:=time*5;
         end;
         myscout[k].IS_HOT    :=TRUE;
         sound(1500);delay(40);
         sound(1600);delay(10);
         sound(800);delay(200);
         sound(1600);delay(60);
         nosound;
         read(datalink,myscout[k].subx);
         read(datalink,myscout[k].suby);
         read(datalink,myscout[k].subcourse);
         read(datalink,myscout[k].subspeed);
         read(datalink,myscout[k].cred);
      end else begin
         myscout[k].IS_HOT       :=FALSE;
      end;
      readln(datalink);
   end;
end;


procedure TempUpdateNegativeInfo(s:byte);
var
i        :integer;
cdp5     :real;
d        :real;
begin
   for i:=1 to NUMBEROFTGTS do begin
```

```
      d:=sqrt(distancer(mytarget[i].x,mytarget[i].y,myscout[s].x,myscout[s].y));
      if d < myscout[s].maxscale  then begin
         cdp5:=DetectionEffort(i,s,d,depth);
         mytarget[i].prob:=mytarget[i].prob * (1-cdp5);
      end;
      if s=numberofscouts then begin
         UpdateTargetPosition(i);
         totprob:=totprob + mytarget[i].prob;
      end;
   end;
end;


procedure TempUpdatePositiveInfo(s:byte);
var
i        :integer;
cdp5     :real;
C        :real;
d        :real;
radweight:real;
distoff  :real;
error    :real;
begin
   for i:=1 to NUMBEROFTGTS do begin
      d:=sqrt(distancer(mytarget[i].x,mytarget[i].y,myscout[s].x,myscout[s].y));
      cdp5:=DetectionEffort(i,s,d,depth);
      C:=myscout[s].cred;
      distoff:=sqrt(distancer(mytarget[i].x,mytarget[i].y,myscout[s].subx,myscout[s].suby));
      error:= d * myscout[s].beamwidth * pi /180;
      radweight:=exp(-sqr(distoff/error)/2);
      mytarget[i].prob:= mytarget[i].prob * (1-C)
                        +mytarget[i].prob * cdp5 * radweight*C;
      TargetReaction(i,s,d);
      if s=numberofscouts then begin
         UpdateTargetPosition(i);
         totprob:=totprob + mytarget[i].prob;
      end;
   end;
end;


procedure UpdatePriorWithScoutsInfo;
var
scout       :integer;
begin
   totprob:=0.0000000000001;

   for scout:=1 to numberofscouts do begin
      if odd(time) then begin
         UpdateNL(scout);
      end;
      if not myscout[scout].IS_HOT then begin
         TempUpdateNegativeInfo(scout);
      end else begin
         TempUpdatePositiveInfo(scout);
      end;
   end;
end;


procedure ResetUndetectedMass;
begin
```

```
undetectedmass:=1;
end;


procedure UpdateUndetectedMass;
begin
undetectedmass:=undetectedmass*totprob;
end;


begin
end.
```

APPENDIX F. DETECMANAGER UNIT

```
unit DetecManager;

interface
uses
CRT,Intermanager,scoutmanager;

function  DetectionEffort(id,sd :integer;distts,dp:real):real;
function  AbsorptionCoefficient(f:real):real;
function  UpdateNL(sc:byte):integer;
function  ScatteringArea(tau,theta:real):integer;
function  ScatteringStrength(svtprofile:integer;freq:real;winspeed,bottom:integer):integer;
function  distancer(x1,y1,x2,y2:real):real;
procedure UpdateSonarParameters;

implementation

{ TOOLS}
{-------------------------------------------------------------------------------}

function distancer(x1,y1,x2,y2:real):real;
var d1,d2:real;
begin
d1:=(x1-x2)*(x1-x2);
d2:=(y1-y2)*(y1-y2);
distancer:=d1+d2;
end;

function  power2( base     : real;
                  exponent : real  ) : real;
{base can not be negative}
begin
power2:= exp( exponent * ln( base ) );
end;  { function power }


function normalcdfwashb(mu,sig,x:real):real;
{based in a program by Alan Washburn}
const
A=0.319381530; B=-0.356563782; C=1.781477937; D=-1.821255978;
E=1.330274429; G= 0.231641900;
var
y,f,p    :real;
positive :boolean;
begin
   if sig<=0 then sig:=1;
   x:=(x-mu)/sig;
   positive:=(x>=0);
   x:=abs(x);
   if x>=100 then x:=100;
   y:=1/(1+G*x);
   f:=C+y*(D+y*E);
   p:=1-exp(-0.5*x*x)/sqrt(8*arctan(1))*y*(A+y*(B+y*f));
   if positive then begin
      normalcdfwashb:=p;
   end else begin
      normalcdfwashb:=1-p;
   end;
end;
```

```
function sumdB(a,b:real):real;
begin
   sumdB:=(10/ln(10))*ln(   power2(10,a/10)  +   power2(10,b/10)  );
end;


{SCATTERING METHODS}
{-------------------------------------------------------------------------------}

function TargetStrength(tgx,tgy,scx,scy,tgc:real):integer;
const
LOWTS=10;
INCTS=8;
var
cospsisqr        :real;
num,den          :real;
tstemp           :integer;
begin
   if (scx <> tgx) or (scy<> tgy) then begin
      num:=sqr((scx-tgx)*sin(tgc*pi/180)+(scy-tgy)*cos(tgc*pi/180));
      den:=((scx-tgx)*(scx-tgx)+(scy-tgy)*(scy-tgy));
      cospsisqr:=num/den;

      tstemp:=round(LOWTS+INCTS*(1-cospsisqr));
      TargetStrength:=tstemp;
      end else begin
      TargetStrength:=LOWTS;
   end;
end;


function  ScatteringArea(tau,theta:real):integer;
const
C=1500;
begin
   theta:=theta*pi/180;
   ScatteringArea:=round( (10/ln(10) ) * ln(tau*theta*C/2)  );
end;


function SurfaceScattering(fr:real;wind:integer):integer;
const
THETA=37*pi/180;
var
h        :real;
begin
   h:=power2(wind,2.5)*0.026;
   SurfaceScattering:= round((10/ln(10)) * power2(ln(fr*h*sin(THETA)), 0.99) - 45.3);
end;


function BottomScattering(fr:real;bot:integer):integer;
begin
   case bot of
      1   : BottomScattering:=round( 0.0120*fr*fr*fr + 0.0283*fr*fr +0.0062*fr - 28.153);
      2   : BottomScattering:=round( 0.0076*fr*fr*fr - 0.2337*fr*fr +2.8700*fr - 40.410);
      2   : BottomScattering:=-18;
   end;
end;
```

```
function  ScatteringStrength(svtprofile:integer;freq:real;winspeed,bottom:integer):integer;
{svtprofile=1 (positive velocity gradient)}
{svtprofile=2 (negative velocity gradient)}
begin
   case svtprofile of
      1 : ScatteringStrength:=SurfaceScattering(freq,winspeed);
      2 : ScatteringStrength:=BottomScattering(freq,bottom);
   end;
end;

procedure UpdateSonarParameters;
var
i:integer;
begin

   for i:=1 to numberofscouts do begin
     myscout[i].reverbarea:=ScatteringArea(myscout[i].pulselength,
                    myscout[i].beamwidth);
     myscout[i].SS:=ScatteringStrength(svtprofiletype,
                    myscout[i].frequency,wind,bottomtype);
   end;
end;



{NOISE METHODS}
{-------------------------------------------------------------------------------}


function AmbientNoise(f:real;w:integer):real;
begin
   AmbientNoise:=46.121+2.221*w-0.041*w*w-17.012*( ln(f)/ln(10) - 3 );
end;


function SelfNoise(v:real;f:real):real;
begin
   SelfNoise:= 23+2.02*v+(20/ln(10))*ln(25/f);
end;

function  NoiseLevel(scoutvel:real;freq:real;windspeed:integer):integer;
var
An             :real;
Sn             :real;
begin
   An:=AmbientNoise(freq,windspeed);
   Sn:=SelfNoise(scoutvel,freq);
   NoiseLevel:=round(sumdB(An,Sn));
end;


function UpdateNL(sc:byte):integer;
begin
   with myscout[sc] do begin
      NL:= NoiseLevel(speed,frequency,wind);
   end;
end;
```

```
{PROPAGATION METHODS}
{-------------------------------------------------------------------------------}

function  AbsorptionCoefficient(f:real):real;
begin
   AbsorptionCoefficient:=    8e-3 /( (0.7 /f*f)+1 )
                           +  0.04/((6000/f*f)+1)    +  4e-7 * f*f ;
end;


function TransLossesGeom(d,H:real):real;
var
TLG     :real;
begin
   if d <= 0 then begin
      TLG:=1;
      end else begin
      TLG:=(10/ln(10))*(ln(d)+ln(H/pi));
   end;
   TransLossesGeom:=TLG;
end;



{DETECTION MODEL METHODS}
{-------------------------------------------------------------------------------}

function  DetectionEffort(id,sd:integer;distts,dp:real):real;
var
p         :real;
TL        :real;
TLGEOM    :real;
FOM       :real;
SE        :real;
TS        :real;
SErevbr   :real;
SEnoise   :real;
tx        :integer;
ty        :integer;
sx        :integer;
sy        :integer;
tc        :real;

begin
   tx:=mytarget[id].x;
   ty:=mytarget[id].y;
   sx:=myscout[sd].x;
   sy:=myscout[sd].y;
   tc:=mytarget[id].course;
   distts:=185*distts;{185 accounts for the scale factor}
   TLGeom:= TransLossesGeom(distts,dp);
   TL:=TLGeom+distts*myscout[sd].abscoeff;
   TS:= TargetStrength(tx,ty,sx,sy,tc);
   FOM:= myscout[sd].SL-myscout[sd].NL-myscout[sd].DT+TS+myscout[sd].DI;
   SEnoise:=FOM-2*TL;
   SErevbr:=TS-TLGeom-myscout[sd].DT-myscout[sd].reverbarea-myscout[sd].SS;
   if SErevbr < SEnoise then begin
      SE:= SErevbr;
      end else begin
      SE:= SEnoise;
   end;
```

```
   p:=power2(1-normalcdfwashb(0, NOISESIGMA,SE), NUMOFPULSES);
   DetectionEffort:=1-p;
end;


begin
end.
```

APPENDIX G. CELLSMANAGER UNIT

```
unit CellsManager;

interface

uses
GRAPH, CRT ;


type
probcelltype = array[1..80,1..80] of single;



var
probincell      :probcelltype;
maxprob         :real;
numberout       :integer;

procedure UpdateInCellProbabilities;
procedure UpdateOutAreaProbabilities;
procedure WriteProbinCell;


implementation
uses GraphManager, Targtmanager,InterManager;


procedure Clearallcells;
var
i,j        :integer;
begin
for i:=1 to MAXGRID do
    begin
    for j:=1 to MAXGRID do
        begin
         probincell[i,j]:=0;
    end;
end;
end;


procedure UpdateInCellProbabilities;
var
i   :integer;
begin
maxprob:=0;
numberout:=0;
ClearallCells;
for i:=1 to NUMBEROFTGTS do
    begin
    if((mytarget[i].x > GRIDSTEP) and (mytarget[i].y > GRIDSTEP )) and
      ((mytarget[i].x <MAXGRID*GRIDSTEP) and (mytarget[i].y <MAXGRID*GRIDSTEP ))
      then
      begin
      probincell[mytarget[i].cellx,mytarget[i].celly]:=
      probincell[mytarget[i].cellx,mytarget[i].celly]+mytarget[i].prob;
      if maxprob < probincell[mytarget[i].cellx,mytarget[i].celly] then
         begin
```

```
        maxprob:= probincell[mytarget[i].cellx,mytarget[i].celly];
        end;
      end else
      begin
      inc(numberout);
    end;
end;
end;


procedure BroadcastProbinCell;
var
i,j        :integer;
begin
assign(outdatalink,"c:\lnk2.dat");
for i:=1 to MAXGRID do
    for j:=1 to MAXGRID do
        writeln(outdatalink,probincell[i,j]);
    end;
end;
end;



begin
end.
```

APPENDIX H.GRAPHMANAGER UNIT

```
unit GraphManager;

interface

uses
GRAPH, CRT;

const
BOXSIZE=100;
SCALECOLORS=10;
BGIdir='..\BGI';
GRIDCOLOR=32;
BASEPATTERN=1;
TEXTCOLOR=8;
TARGETSCOLOR=30;
SCOUTSCOLOR=24;
PALETTE1=11;
PALETTE2=12;
MAXGRID=80;
GRIDSTEP=5;
CENTER=MAXGRID*GRIDSTEP div 2+GRIDSTEP;
STEP          = 5;
FRAME         =80;

type
colortype       = array[1..SCALECOLORS]of byte;
celltype        = array[1..MAXGRID,1..MAXGRID] of integer;

const
COLORCODE     : colortype = (8,1,33,12,4,36,52,62,55,63);



procedure InitGraphics;
procedure SetGrid;
procedure paint(x,y,pat,col:word);
procedure SetScale;
procedure SetTime(t:integer);
procedure DisplayTargets;
procedure DisplayScouts;
procedure PleaseWait;
procedure SetProbDistrColors;
procedure OutputGraph;
procedure Start;
procedure DisplayData;
procedure DisplayContactSummary(srchr:byte);


implementation

uses TargtManager,ScoutManager,CellsManager,Intermanager;


procedure InitGraphics;
var
GraphDriver,
GraphMode,
i         : integer;
```

```
begin
     GraphDriver := GRAPH.Detect;
     GRAPH.InitGraph(GraphDriver,GraphMode,BGIdir);
     for i:=1 to SCALECOLORS do begin
               SetPalette(i,COLORCODE[i]);
     end;
end;{initgraphics}


procedure SetGrid;
var
i        :integer;
oldstyle :TextSettingsType;
begin
GetTextSettings(oldstyle);
SetPalette(15,GRIDCOLOR);
for i:=1 to MAXGRID do begin
    line(i*GRIDSTEP,GRIDSTEP,i*GRIDSTEP,GRIDSTEP*MAXGRID);
    line(GRIDSTEP,i*GRIDSTEP,GRIDSTEP*MAXGRID,i*GRIDSTEP);
end;
SetTextStyle(2,0,(STEP+1) div 2  );
line(GRIDSTEP+MAXGRID*GRIDSTEP div 2,0,GRIDSTEP+MAXGRID*GRIDSTEP div 2,GRIDSTEP);
OutTextXY(MAXGRID*GRIDSTEP div 2,0,chr(48));
OutTextXY(0,MAXGRID*GRIDSTEP div 2,chr(48));
OutTextXY(MAXGRID*GRIDSTEP div 4,0,concat(chr(49),chr(48)));
OutTextXY(0,MAXGRID*GRIDSTEP div 4,concat(chr(49),chr(48)));
OutTextXY(MAXGRID*GRIDSTEP*3 div 4,0,concat(chr(49),chr(48)));
OutTextXY(0,MAXGRID*GRIDSTEP*3 div 4,concat(chr(49),chr(48)));
OutTextXY(MAXGRID*GRIDSTEP-GRIDSTEP,0,concat(chr(50),chr(48)));
OutTextXY(0,0,concat(chr(50),chr(48)));
OutTextXY(0,MAXGRID*GRIDSTEP-GRIDSTEP,concat(chr(50),chr(48)));
SetColor(15);
end;


procedure paint(x,y,pat,col:word);
begin
SetFillStyle(pat, col);
FloodFill(x,y, GetMaxColor);
end;

procedure SetScale;
var
i        :integer;
outstring:string;
begin
line(STEP,FRAME*STEP+STEP*3,
     STEP*2*SCALECOLORS+STEP,FRAME*STEP+STEP*3);
line(STEP,FRAME*STEP+STEP*5,
     STEP*2*SCALECOLORS+STEP,FRAME*STEP+STEP*5);
for i:=1 to SCALECOLORS+1 do
    begin
    line(i*2*STEP-STEP,FRAME*STEP+STEP*3,
         i*2*STEP-STEP,FRAME*STEP+STEP*5);
end;
for i:=1 to SCALECOLORS do
    begin
    paint(i*2*STEP,FRAME*STEP+STEP*4,BASEPATTERN,i);
    SetTextStyle(2,0,(STEP+2) div 2 );
    OutTextXY(i*2*STEP-10,FRAME*STEP+STEP,
              concat(chr(46),chr(47+i)));
```

```
end;
OutTextXY((SCALECOLORS+1)*2*STEP-STEP,FRAME*STEP+STEP,chr(49));
end;


procedure DisplayTargets;
var
i       :integer;
begin
SetPalette(PALETTE1,TARGETSCOLOR);
for i:=1 to NUMBEROFTGTS do
    begin
    if ((mytarget[i].x >GRIDSTEP) and (mytarget[i].x <MAXGRID*GRIDSTEP)) and
       ((mytarget[i].y >GRIDSTEP) and (mytarget[i].y <MAXGRID*GRIDSTEP)) then
                    PutPixel(mytarget[i].x,mytarget[i].y,PALETTE1);
end;
end;


procedure DisplayScouts;
var
i       :integer;
begin
SetPalette(PALETTE2,SCOUTSCOLOR);
for i:=1 to numberofscouts do
    begin
    if ((myscout[i].x >GRIDSTEP) and (myscout[i].x <MAXGRID*GRIDSTEP)) and
       ((myscout[i].y >GRIDSTEP) and (myscout[i].y <MAXGRID*GRIDSTEP)) then
       begin
       PutPixel(myscout[i].x,myscout[i].y,PALETTE2);
       SetColor(PALETTE2);
       OutTextXY(myscout[i].x + 5, myscout[i].y - 5, concat(myscout[i].id
                ,chr(48),chr(48),chr(48+i)));
       Circle(myscout[i].x,myscout[i].y,myscout[i].maxscale);
    if myscout[i].IS_HOT then begin
       PutPixel(myscout[i].subx+5, myscout[i].suby+5,8);
       PutPixel(myscout[i].subx+6, myscout[i].suby+4,8);
       PutPixel(myscout[i].subx+7, myscout[i].suby+3,8);
       PutPixel(myscout[i].subx+8, myscout[i].suby+2,8);
       PutPixel(myscout[i].subx+9, myscout[i].suby+1,8);
       PutPixel(myscout[i].subx+10, myscout[i].suby+0,8);
       PutPixel(myscout[i].subx+11, myscout[i].suby-1,8);
       PutPixel(myscout[i].subx+4, myscout[i].suby+4,8);
       PutPixel(myscout[i].subx+3, myscout[i].suby+3,8);
       PutPixel(myscout[i].subx+2, myscout[i].suby+2,8);
       PutPixel(myscout[i].subx+1, myscout[i].suby+1,8);
       PutPixel(myscout[i].subx+0, myscout[i].suby+0,8);
       PutPixel(myscout[i].subx-1, myscout[i].suby-1,8);
       SetColor(14);
       OutTextXY(myscout[i].subx +5, myscout[i].suby-10, concat(chr(48+i)));
       SetColor(15);

    end;
    end;
end;
end;


procedure SetProbDistrColors;
var
i,j,c   :integer;
begin
```

```
if maxprob > 0 then begin
   for i:=1 to MAXGRID-1 do begin
      for j:=1 to MAXGRID-1 do begin
            for c:=0 to SCALECOLORS-1 do begin
          if (100*probincell[i,j]/maxprob>c*SCALECOLORS)
          and (100*probincell[i,j]/maxprob<=(c*SCALECOLORS+100/SCALECOLORS)) then begin
             paint(i*GRIDSTEP+1,j*GRIDSTEP+1,BASEPATTERN,c+1);
                    end;
             end;
      end;
   end;
end else begin
   OutTextXY(40,40,'No target in the area,it is gone');
end;
end;


procedure PleaseWait;
begin
InitGraphics;
outtextxy(MAXGRID*GRIDSTEP+10, GRIDSTEP, 'Please wait..');
end;


procedure DisplayLogo;
var
i        :integer;
MAXx     :integer;
MAXy     :integer;
begin
SetPalette(15,GRIDCOLOR);
MAXx:=GetMaxX;
MAXy:=GetMaxY;
line(round((MAXx/2)-BOXSIZE),round((MAXy/2)-BOXSIZE),
     round((MAXx/2)+BOXSIZE),round((MAXy/2)-BOXSIZE));
line(round((MAXx/2)-BOXSIZE),round((MAXy/2)+BOXSIZE),
     round((MAXx/2)+BOXSIZE),round((MAXy/2)+BOXSIZE));
line(round((MAXx/2)-BOXSIZE),round((MAXy/2)-BOXSIZE),
     round((MAXx/2)-BOXSIZE),round((MAXy/2)+BOXSIZE));
line(round((MAXx/2)+BOXSIZE),round((MAXy/2)-BOXSIZE),
     round((MAXx/2)+BOXSIZE),round((MAXy/2)+BOXSIZE));
line(round((MAXx/2)-BOXSIZE),round((MAXy/2)+BOXSIZE+5),
     round((MAXx/2)+BOXSIZE),round((MAXy/2)+BOXSIZE+5));
line(round((MAXx/2)-BOXSIZE),round((MAXy/2)+BOXSIZE+25),
     round((MAXx/2)+BOXSIZE),round((MAXy/2)+BOXSIZE+25));
for i:=0 to SCALECOLORS do
    begin
    line(round((MAXx/2)-BOXSIZE)+i*round(BOXSIZE/5),round((MAXy/2)+BOXSIZE+5),
         round((MAXx/2)-BOXSIZE)+i*round(BOXSIZE/5),round((MAXy/2)+BOXSIZE+25));
    end;
for i:=1 to SCALECOLORS-1 do
    begin
    paint(round((MAXx/2)-BOXSIZE)+1+i*round(BOXSIZE/5),round((MAXy/2)+BOXSIZE+6),
          BASEPATTERN+8,i);
    delay(160);
    end;
SetPalette(1,8);
paint(round(MAXx/2),round(MAXy/2),BASEPATTERN,1);
SettextStyle(1,0,1);
OuttextXY(round(MAXx/2)-50,round(MAXy/2)-50,'  NRTCAS ');
SettextStyle(2,0,4);
OuttextXY(round(MAXx/2)-100,round(MAXy/2)-25,   '        Reactive Target');
```

```
OuttextXY(round(MAXx/2)-100,round(MAXy/2)-15   ,' Computer Assisted Passive Search');
SettextStyle(2,0,4);
OuttextXY(round(MAXx/2)-100,round(MAXy/2),      '           Release 1.1 ');
OuttextXY(round(MAXx/2)-100,round(MAXy/2)+35,      '              NPS  ');
SettextStyle(2,0,3);
OuttextXY(round(MAXx/2)-100,round(MAXy/2)+70,     '                  C.RECALDE ');
end;


procedure DisplayIntro;
var
MAXx     :integer;
MAXy     :integer;
begin
cleardevice;
SetPalette(15,GRIDCOLOR);
MAXx:=GetMaxX;
MAXy:=GetMaxY;
line(GRIDSTEP,GRIDSTEP,MAXx-GRIDSTEP,GRIDSTEP);
line(GRIDSTEP,GRIDSTEP,GRIDSTEP,MAXy-GRIDSTEP);
line(MAXx-GRIDSTEP,MAXy-GRIDSTEP,GRIDSTEP,MAXy-GRIDSTEP);
line(MAXx-GRIDSTEP,MAXy-GRIDSTEP,MAXx-GRIDSTEP,GRIDSTEP);
SetColor(TEXTCOLOR-2);
SetTextStyle(1,0,1);
SetTextJustify(1,1);
outtextxy(round(MAXx/2),GRIDSTEP+30,
'        Non Reactive Target  Computer Assisted Search    ');
SetTextStyle(2,0,4);
outtextxy(round(MAXx/2),GRIDSTEP+90,
'    This program will perform a  Montecarlo Simulation to assist in ');
outtextxy(round(MAXx/2),GRIDSTEP+100,
'    conduction of a Lost Contact Search of a Non Reactive Target by ');
outtextxy(round(MAXx/2),GRIDSTEP+110,
'    Passive Searchers');
outtextxy(round(MAXx/2),GRIDSTEP+130,
'    The user must provide parameters regarding:                    ');
outtextxy(round(MAXx/2),GRIDSTEP+150,
'                    THE TARGET                                     ');
outtextxy(round(MAXx/2),GRIDSTEP+160,
'                    THE SEARCHER                                   ');
outtextxy(round(MAXx/2),GRIDSTEP+170,
'                    THE SIMULATION                                 ');
outtextxy(round(MAXx/2),GRIDSTEP+210,
'    The program will output a Probability Distribution Map in  Color');
outtextxy(round(MAXx/2),GRIDSTEP+220,
'    Codes relative to the  maximum value of  probability encountered');
outtextxy(round(MAXx/2),GRIDSTEP+230,
'    in any of the  square cells in wich the Local Area is divided.  ');
outtextxy(round(MAXx/2),GRIDSTEP+300,
'    Please, press any key to continue');
readkey;
end;


procedure Start;
begin
clrscr;
InitGraphics;
DisplayLogo;
{delay(4000);}
DisplayIntro;
```

```
Closegraph;
end;


procedure SetTime(t:integer);
var
fstdigit      :byte;
snddigit      :byte;
trddigit      :byte;
begin
t:=t*5;
SetTextStyle(2,0,5);
OutTextXY(STEP,FRAME*STEP+STEP*5, 'time  : ');
SetColor(TEXTCOLOR-2);
fstdigit:=round(t div 100);
snddigit:=round((t mod 100) div 10) ;
trddigit:=round((t mod 100) mod 10) ;
OutTextXY(STEP+STEP*11,FRAME*STEP+STEP*5,
          concat(chr(48+fstdigit),chr(48+snddigit),chr(48+trddigit)));
end;

procedure OutputResults;
var
MAXx     :integer;
MAXy     :integer;
fstdigit      :byte;
snddigit      :byte;
trddigit      :byte;
fthdigit      :byte;
fihdigit      :byte;
p             :real;
count         :byte;
holder        :integer;
stringholder  :string;

begin
MAXx:=GetMaxX;
MAXy:=GetMaxY;
SetPalette(15,GRIDCOLOR);
Setcolor(15);
line(FRAME*STEP+5,STEP,MAXx-10,STEP);
line(FRAME*STEP+5,STEP,FRAME*STEP+5,MAXy-10);
line(MAXx-10,MAXy-10,STEP*FRAME+5,MAXy-10);
line(MAXx-10,MAXy-10,MAXx-10,STEP);
SetColor(TEXTCOLOR-2);
OutTextXY(FRAME*STEP+15,STEP*3,'      RTCAS STATUS ');
SetColor(15)        ;
SetTextStyle(2,0,4);
OutTextXY(FRAME*STEP+15,STEP*28,'UNDET PROB MASS ');
p:=undetectedmass*10000;
fstdigit:=round(p) div 10000;
snddigit:=(round(p) mod 10000)div 1000;
trddigit:=((round(p) mod 10000)mod 1000)div 100;
fthdigit:=(((round(p) mod 10000)mod 1000)mod 100)div 10;
fihdigit:=(((round(p) mod 10000)mod 1000)mod 100)mod 10;
SetColor(TEXTCOLOR-2);
OutTextXY(FRAME*STEP+STEP*30,STEP*28,
          concat(chr(48+fstdigit),chr(46),chr(48+snddigit),
          chr(48+trddigit),chr(48+fthdigit),chr(48+fihdigit)));

SetColor(15)       ;
SetTextStyle(2,0,4);
```

```
OutTextXY(FRAME*STEP+15,STEP*32,'Number of TGTS OFF  ');
fstdigit:=round(numberout div 100);
snddigit:=round((numberout mod 100) div 10) ;
trddigit:=round((numberout mod 100) mod 10) ;
SetColor(TEXTCOLOR-2);
OutTextXY(FRAME*STEP+STEP*30,STEP*32,
          concat(chr(48+fstdigit),chr(48+snddigit),chr(48+trddigit)));

SetPalette(15,GRIDCOLOR);
setcolor(15);
line(FRAME*STEP+5,STEP*40,MAXx-10,STEP*40);

SetTextStyle(2,0,4);
SetColor(TEXTCOLOR-2);
OutTextXY(FRAME*STEP+15,STEP*42,'       MC SIMULATION INFO');

SetColor(15);
OutTextXY(FRAME*STEP+15,STEP*45,'TARGET DATA');
OutTextXY(FRAME*STEP+15,STEP*48,'DATUM ERROR (nm)  ');
fstdigit:=round(sigmarinit div 100);
snddigit:=round((sigmarinit mod 100) div 10) ;
trddigit:=round((sigmarinit mod 100) mod 10) ;
SetColor(TEXTCOLOR-2);
OutTextXY(FRAME*STEP+STEP*30,STEP*48,
          concat(chr(48+fstdigit),chr(48+snddigit),
                      chr(48+trddigit)));

SetColor(15);
OutTextXY(FRAME*STEP+15,STEP*50,'REACTION RANGE (nm)  ');
fstdigit:=round(reactrange div 100);
snddigit:=round((reactrange mod 100) div 10) ;
trddigit:=round((reactrange mod 100) mod 10) ;
SetColor(TEXTCOLOR-2);
OutTextXY(FRAME*STEP+STEP*30,STEP*50,
          concat(chr(48+fstdigit),chr(48+snddigit),
                      chr(48+trddigit)));


SetColor(15);
OutTextXY(FRAME*STEP+15,STEP*52,'REACTION TACK  ');
fstdigit:=round(reactiontack div 100);
snddigit:=round((reactiontack mod 100) div 10) ;
trddigit:=round((reactiontack mod 100) mod 10) ;
SetColor(TEXTCOLOR-2);
OutTextXY(FRAME*STEP+STEP*30,STEP*52,
          concat(chr(48+fstdigit),chr(48+snddigit),
                      chr(48+trddigit)));


SetColor(15);
OutTextXY(FRAME*STEP+15,STEP*54,'MIN SPEED          ');
fstdigit:=round(minsubspeed div 100);
snddigit:=round((minsubspeed mod 100) div 10) ;
trddigit:=round((minsubspeed mod 100) mod 10) ;
SetColor(TEXTCOLOR-2);
OutTextXY(FRAME*STEP+STEP*30,STEP*54,
          concat(chr(48+fstdigit),chr(48+snddigit),
                      chr(48+trddigit)));

SetColor(15);
OutTextXY(FRAME*STEP+15,STEP*56,'MAX SPEED         ');
```

```
fstdigit:=round(maxsubspeed div 100);
snddigit:=round((maxsubspeed mod 100) div 10) ;
trddigit:=round((maxsubspeed mod 100) mod 10) ;
SetColor(TEXTCOLOR-2);
OutTextXY(FRAME*STEP+STEP*30,STEP*56,
         concat(chr(48+fstdigit),chr(48+snddigit),
                        chr(48+trddigit)));

SetColor(15);
line(FRAME*STEP+5,STEP*60,MAXx-10,STEP*60);
SetColor(TEXTCOLOR-2);
OutTextXY(FRAME*STEP+15,STEP*62,'     REAL WORLD SEARCH INFO ');
SetColor(15);
OutTextXY(FRAME*STEP+15,STEP*65,'SEARCHERS DATA');
OutTextXY(FRAME*STEP+15,STEP*69,'SCOUT  COURSE SPEED SONAR HOT ');
for count:=1 to 8 do begin
   if count <= numberofscouts then begin
      SetColor(TEXTCOLOR-2);
      OutTextXY(FRAME*STEP+15,STEP*70+count*2*STEP,   concat(
                myscout[count].id,chr(48),chr(48),chr(48+count),' '));
      fstdigit:=round(myscout[count].course div 100);
      snddigit:=round((myscout[count].course mod 100) div 10) ;
      trddigit:=round((myscout[count].course mod 100) mod 10) ;
      OutTextXY(FRAME*STEP+55,STEP*70+count*2*STEP,concat(
                chr(48+fstdigit),chr(48+snddigit),chr(48+trddigit)));
      holder:=round(myscout[count].speed);
      fstdigit:=round(holder div 100);
      snddigit:=round((holder mod 100) div 10) ;
      trddigit:=round((holder mod 100) mod 10) ;
      OutTextXY(FRAME*STEP+95,STEP*70+count*2*STEP,concat(
                chr(48+fstdigit),chr(48+snddigit),chr(48+trddigit)));
      if myscout[count].SONAR_ON then begin
         stringholder:=' ON ';
      end else begin
         stringholder:=' OFF';
      end;
      OutTextXY(FRAME*STEP+135,STEP*70+count*2*STEP,stringholder);
{***} myscout[2].IS_HOT:=TRUE;
      if myscout[count].IS_HOT then begin
         stringholder:=' YES';
         SetColor(14);
         OutTextXY(FRAME*STEP+175,STEP*70+count*2*STEP,stringholder);
      end;
      if myscout[count].initialhot <> 0 then begin
         DisplayContactSummary(count);
      end;
   end;
end;
SetTextStyle(2,0,4);
SetColor(15);
OutTextXY(FRAME*STEP+15,MAXy-30,'Hit any key to go back to main menu');
end;{procedure}



procedure DisplayContactSummary(srchr:byte);
var
holder:integer;
fstdigit:integer;
snddigit:integer;
trddigit:integer;
```

```
begin
SetColor(TEXTCOLOR-2);
OutTextXY(30*STEP,FRAME*STEP+STEP,'INITIAL CONTACT REPORTS ');
SetColor(15);
OutTextXY(30*STEP,FRAME*STEP+STEP*3,'SEAR  TIME  COURSE  SPEED ');
SetColor(14);
OutTextXY(30*STEP,FRAME*STEP+STEP*3+STEP*srchr*2,concat(
        myscout[srchr].id,chr(48),chr(48),chr(48+srchr)));
fstdigit:=round(myscout[srchr].initialhot div 100);
snddigit:=round((myscout[srchr].initialhot mod 100) div 10) ;
trddigit:=round((myscout[srchr].initialhot mod 100) mod 10) ;
OutTextXY(30*STEP+35,FRAME*STEP+STEP*3+STEP*srchr*2,concat(
        chr(48+fstdigit),chr(48+snddigit),chr(48+trddigit)));
fstdigit:=round(myscout[srchr].subcourse div 100);
snddigit:=round((myscout[srchr].subcourse mod 100) div 10) ;
trddigit:=round((myscout[srchr].subcourse mod 100) mod 10) ;
OutTextXY(30*STEP+80,FRAME*STEP+STEP*3+STEP*srchr*2,concat(
        chr(48+fstdigit),chr(48+snddigit),chr(48+trddigit)));
holder:=round(myscout[srchr].subspeed);
fstdigit:=round(holder div 100);
snddigit:=round((holder mod 100) div 10) ;
trddigit:=round((holder mod 100) mod 10) ;
OutTextXY(30*STEP+125,FRAME*STEP+STEP*3+STEP*srchr*2,concat(
        chr(48+fstdigit),chr(48+snddigit),chr(48+trddigit)));
end;


procedure DisplayData;
begin
ClearDevice;
SetGrid;
SetScale;
DisplayTargets;
DisplayScouts;
undetectedmass:=1;
OutputResults;
readkey;
cleardevice;
CloseGraph;
end;



procedure OutputGraph;
begin
ClearDevice;
SetGrid;
SetScale;
SetTime(totalsimtime);
{DisplayTargets;}
SetProbDistrColors;
DisplayScouts;
OutputResults;
readkey;
ClearDevice;
CloseGraph;
end;

begin
end.
```

LIST OF REFERENCES

Borland International Inc.,*Turbo Pascal Version 7.0*, 1992.

Buss, Arnold H., Stork, Kirk, *Discrete Event Simulation on the World Wide Web Using Java*, To appear in Proceedings of the 1996 Winter Simulation Conference, 1996.

Cheong, Edmond, *A Zero Sum Helicopter Submarine Game*, Master'S Thesis, Naval Postgraduate School, 1994.

Devore, Jay, *Probability and Statistics for Engineering and the Science*, Duxbury Press, 1994.

Dudewicz, Eduard J., Mishra, Satya N., *Modern Mathematical Statistics*, John Wiley & Sons, 1988.

Elliot, Steven D., Miller,Phillip L., *Inside 3D Studio*, Release 4, New Riders Publishing,1995.

Hughes, Wayne P., *Tactica de Flota*, Instituto de Publicaciones Navales, Buenos Aires,1988.

Hughes, Wayne P., *A Perspective on Joint Litoral Warfare*, Study for the Applied Physics Laboratory, John Hopkins University, Washington, 1993.

Kinsler Laurence, Frey Austin, Coppens, Alan, Sanders, James, *Fundamentals of Acoustics*, Wiley & Sons, 1980.

Stone Lawrence, Corwin Thomas, *Technical Documentation of Nodestar*, Metron Inc., 1995.

Stone Lawrence, *A comparison of Nodestar and MTST*, Metron Inc., 1995.

Sun Microsystems, *Java Programming Language Version 1.0*, 1994.

Urick, Robert, *Principles of Underwater Sound*, McGraw Hill, 1975.

Urick, Robert, *Reverberation-Derived Scattering Strength of the Shallow Sea Bed*, Journal of the Acoustic Society of America,1970.

Wagner, Daniel, *Naval Tactical Decision Aids*, Lecture Notes, Naval Postgraduate School, 1989.

Washburn, Alan R., *Search and Detection*, Operations Research Society of America Books, 1989.

Washburn, Alan R., *Notes on Cumulative Detection Probability*, Lecture Notes, Naval Postgraduate School, 1995.

Washburn, Alan R., *Multiple Experts and Credibility in Search TDA's*, Lecture Notes, Naval Postgraduate School, 1995.

Widdis, Daniel, *Using Negative Inforamtion to Improve Performance of Froward Scatter Arrays*, Master's Thesis, Naval Postgraduate School, 1995.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center 8725 John J. KingmanRoad., Ste 0944 Ft. Belvoir, VA 22060-6218	2
2.	Dudley Knox Library Naval Postgraduate School 411 Dyer Rd. Monterey, California 93943-5101	2
3.	Departament of Operations Research Attn: Professor Alan Washburn, Code OR/Ws Naval Postgraduate School Monterey, California 93943-5000	1
4.	Departament of Phisics Attn: Professor James Sanders, Code PH/Sd Naval Postgraduate School Monterey, California 93943-5000	1
5.	Undersea Warfare Group Attn: Professor James Eagle, Code UW/Er Naval Postgraduate School Monterey, California 93943-5000	1
6.	Undersea Warfare Group Attn: Professor Don Brutzman, Code UW/Br Naval Postgraduate School Monterey, California 93943-5000	1
7.	Departament of Operations Research Attn: Professor Arnold Buss, Code OR/Bu Naval Postgraduate School Monterey, California 93943-5000	1
8.	Argentine Naval Commission 630 Indiana Avenue NW Washington D.C. 20004-2989	1
9.	Argentine Naval Commission Attn: Comando de Operaciones Navales 630 Indiana Avenue NW Washington D.C. 20004-2989	2

10. Argentine Naval Commission
Attn:Comando de la Flota de Mar
630 Indiana Avenue NW
Washington D.C. 20004-2989 — 2

11. Argentine Naval Commission
Attn:Servicio de Analisis Operativo.
630 Indiana Avenue NW
Washington D.C. 20004-2989 — 2

12. Argentine Naval Commission
Attn:Lt. Cesar Recalde.
630 Indiana Avenue NW
Washington D.C. 20004-2989 — 2

13. Argentine Naval Commission
Attn:Direccion de Instruccion Naval.
630 Indiana Avenue NW
Washington D.C. 20004-2989 — 2